Disseny i implementació d'un optimitzador de rutes marítimes

Facultat de Informàtica de Barcelona

Universitat Politècnica de Catalunya – BarcelonaTech

Grau en enginyeria informàtica

Computació

Autor:
Sardà Campomanes, David
Director:
Larrosa Bondia, Francisco Javier (CS)

Data de defensa:
22 d'abril de 2015

Índex

Índex de figures

Índex de taules

Part I

Resums

Català

En la història de la navegació marítima, l'optimització de rutes ha estat un tema poc tractat per la ciència. La manca d'observacions sistemàtiques de l'estat de la mar, i els escassos avenços en meteorologia, tal i com l'entenem avui en dia, han fet que l'elecció de la ruta d'una embarcació fos un afer exclusiu del capità d'aquesta, i la seva experiència com a tal.

Durant els 50 darrers anys, els avenços en oceanografia i meteorologia han permès obtenir models prou acurats com per a plantejar-se mètodes per a complementar els coneixements dels navegants. Els avenços en informàtica han permès aplicar aquests mètodes, i ajustar-los cada cop més a la realitat, tot i que encara són molt primitius (es limiten a informar de l'estat de la mar en un futur, escollir una ruta a partir d'un conjunt de rutes preestablertes molt reduïdes, etc).

L'objectiu d'aquest projecte és desenvolupar un programa capaç de calcular les diferents rutes possibles entre dos punts navegables qualsevol optimitzant més d'un paràmetre (com per exemple temps, confort, combustible, etc), fet que eleva el problema a resoldre a una dificultat de NP-Difícil. Per tant, es requereix que l'algoritme a implementar sigui molt més sofisticat, i és per això que el resoldrem basant-nos en un dels algoritmes de pathfinding (cerca de rutes) actuals més innovadors, el NAMOA* (2008)[1].

Castellano

En la historia de la navegación marítima, la optimización de rutas ha sido un tema poco tratado por la ciencia. La falta de observaciones sistemáticas del estado de la mar, y los escasos avances en meteorología, tal y como la entendemos hoy en día, han hecho que la elección de la ruta de una embarcación fuese un tema exclusivo del capitán de la misma, y de su experiencia como tal.

Durante los últimos 50 años, los avances en oceanografía y meteorología ha permitido obtener modelos lo suficientemente precisos como para plantearse métodos para complementar los conocimientos de los navegantes. Los avances en informática han permitido aplicar estos métodos, y ajustar-los cada vez mas a la realidad, aunque aún son muy primitivos (se limitan a informar del estado de la mar en un futuro, escoger una ruta a partir de un conjunto de rutas preestablecidas, etc).

El objetivo de este proyecto es el de desarrollar un programa capaz de calcular las diferentes rutas posibles entre dos puntos navegables cualquiera optimizando más de un parámetro (como por ejemplo tiempo, confort, combustible, etc), hecho que eleva el problema a resolver a una dificultad de NP-Difícil. Por tanto, se requiere que el algoritmo a implementar sea mucho mas sofisticado, y es por eso que lo resolveremos basándonos en uno de los algoritmos de pathfinding (búsqueda de rutas) actuales mas innovadores, el NAMOA* (2008)[1].

English

In the history of maritime navigation, route optimization has been a topic not very well investigated. The missing systematic maritime weather forecasts, and the small progress made in meteorology, as we understand it today, has made choosing which route a ship will follow an exclusive matter of the ship's captain and his own experience.

During the past 50 years, the advancements in oceanography and meteorology led to obtaining weather models sufficiently precise as to plan new methods to complement the knowledge of the sailors. The improvements in computer science made able to apply those methods, and adjust them to reality, even if this methods nowadays are still too primitive (they limit themselves to specific maritime weather forecasts, or to choose a route based on a small predefined group of routes, etc).

The objective of this project is to develop a program able to calculate the different possible routes between two naval points optimizing more than one parameter (for example time, comfort, gas, etc), fact that increases the problem difficulty to NP-Hard. That's the reason we require the algorithm to implement to be much more sophisticated, and that's why we will solve it based on one of the most innovative pathfinding algorithms nowadays, NAMOA* (2008)[1].

Part II

Introducció i planificació

Capítol 1

Introducció i estat de l'art

1.1 Introducció

Aquest projecte respon a l'oferta proposada per Javier Larrosa, catedràtic de Ciències de la Computació i actual director del projecte, per realitzar un optimitzador de rutes marítimes. La raó d'aquesta proposta era per la possible aplicació d'un algoritme de pathfinding multi-objectiu molt innovador, el NAMOA*, per a la resolució d'un problema crític conegut, no resolt satisfactòriament, com és el de trobar amb un temps computacional raonable rutes marítimes realistes i eficients entre dos punts navegables qualsevol.

A més, la resolució d'aquest problema és d'interès de diverses empreses del sector de l'enginyeria marítima, en concret es tenia coneixement de l'interès de l'empresa SIMO (que de fet col·labora en l'aportació de les dades marítimes, i en el desenvolupament posterior del visor del programa), i a posteriori, de Puertos del Estado (l'organització oficial dedicada a la coordinació i eficiència dels ports d'Espanya).

Per tant, l'objectiu principal del projecte és el d'aconseguir implementar un programari que donats dos punts navegables qualsevol i una serie de preferències (entre les quals hi ha fins a dos paràmetres a optimitzar, com poden ser temps, combustible, altura d'onatge, etc), obtingui un conjunt de rutes no estrictament superiors entre si (pareto-òptimes) que compleixin les condicions exigides.

Aquest programari implementat permetrà disposar als navegants de rutes realistes actualitzades segons l'estat de la meteorologia i el mar que permetran obtindre una major eficiència en la navegació, aspecte crític en tots els sentits.

La dificultat del problema resideix en la seva pròpia naturalesa, trobar la ruta o

rutes considerades òptimes en l'entorn de la navegació marítima és molt costós ja que, si bé trobar la ruta òptima entre dos punts coneguts en un espai limitat i amb un heurístic consistent és un problema de cost polinòmic, el fet de que s'hagi d'optimitzar més d'un paràmetre converteix el problema en NP-complet, i per tant d'un alt cost computacional.

1.2 Estat de l'art

1.2.1 Historia del pathfinding

Les tècniques de pathfinding son la base de la resolució del problema plantejat en aquest projecte. De forma resumida, les tècniques de pathfinding consisteixen en buscar en un graf G, donat un node inicial n pertanyent al mateix graf G, el camí fins a un altre node destí m mitjançant la exploració de nodes adjacents, generalment amb la intenció de trobar el camí mes curt o camí òptim.

Els orígens del pathfinding son difícils de concretar [2], principalment degut a que es un problema que es moderadament senzill de resoldre en condicions senzilles i conegudes, i per tant ha sigut relegat majoritàriament a la intuïció i experiència.

Es precisament aquest caràcter intuïtiu del problema el que pot explicar que quant el problema de trobar la ruta òptima entre dos punts d'un graf va atreure el focus d'atenció en la dècada dels 50, diversos investigadors van arribar a resultats similars de forma independent.

No obstant, el problema va presentar certes dificultats des de bon principi, i va passar un cert temps en que es presentaven solucions subòptimes al problema (com per exemple Shimbel (1955) [3] o Rosenfeld (1956) [4]).

Poc mes tard, van sorgir els primers dos algoritmes de pathfinding bàsics més coneguts avui en dia, l'algoritme de Bellman-Ford (1958) [5] [6] i el de Dijkstra (1959) [7].

Aquests algoritmes eren capaços de trobar tots els camins més curts des d'un vertex fins tots els restants d'un graf ponderat, amb la diferencia de que Bellman-Ford

admetia arestes amb pesos *negatius*, mentres que Dijkstra no. No obstant, Djkstra va resultar ser força més ràpid, sent el temps de calcul en el cas pitjor de Dijkstra de l'ordre de $O(N^2)$, mentres que Bellman-Ford te un cost en el cas pitjor de $O(N*M)$, sent N el número de nodes del graf i M el número de arestes (el n^o *d'arestes* és >= al n^o *de nodes* $-$ 1 sempre que no hi hagi cap node desconnectat, i normalment es molt superior).

Aquests dos algoritmes van resoldre el problema del pathfinding de forma molt eficient, i van ser la base dels algoritmes posteriors, generalment basats en l'algoritme de Dijkstra, que es van implementar.

El mes conegut d'aquests algoritmes es l'A* (1968) [8], un algoritme que es basa en l'ús de funcions heurístiques per tal de reduir el temps de càlcul. A més, A* tan sols s'encarrega de trobar el camí entre dos nodes concrets, i no entre un node i la resta com fa Dijkstra (de fet Dijkstra es podria considerar un cas concret de l'algoritme A* on $h(x) = 0$ per cada node x (on $h(node)$ es la funció heurística)).

Per altra banda, A* es l'algoritme base del que parteixen la majoria dels algoritmes moderns de pathfinding. Aquestes versions, en comptes de intentar millorar la eficiència del A* en general, son més bé adaptacions del algoritme a casos concrets. Per exemple, IDA* (1985) [9] que es una versió del A* adaptada a entorns on es disposi de molt poca memòria, o D* (1994) [10] que es una versió adaptada al camp de la robòtica per cerques repetides en un mateix graf.

1.2.2 Models de predicció meteorològica actuals

Un altre dels elements fonamentals d'aquest projecte son les dades meteorològiques en les que es basa. La obtenció de les mateixes ha sigut basada en tècniques desenvolupades recentment.

Des de la segona meitat del segle XX, el ràpid creixement en el coneixement dels processos atmosfèrics i hidrosferics (en gran part pels avenços en computació) ha proveït als meteoròlegs de les eines per a la previsió del clima i la meteorologia de forma precisa. De fet, gracies a aquests avenços es va començar a incorporar

tècniques de les prediccions meteorològiques en operacions marítimes.

La navegació marítima anterior al 1990 ja utilitzava previsions meteorològiques, la precisió de les quals no ha parat d'augmentar, però la interpretació d'aquestes previsions i presa de decisions era, malgrat tot, una tasca que depenia exclusivament de l'habilitat humana, de la tripulació. Un capità, per exemple, era capaç d'estimar una ruta donades unes condicions meteorològiques, basant-se en la seva experiència i la fiabilitat de les previsions. Motte (1972) [11] i Bowditch (1995) [12] ofereixen una perspectiva històrica sobre aquest aspecte. Malgrat això, a la dècada dels 90, el fort increment en la precisió i resolució dels models geofísics, acompanyat de l'increment en la capacitat computacional dels sistemes informàtics, va encoratjar els científics a desenvolupar models numèrics dedicats a aplicacions específiques amb alta resolució en l'espai i el temps. Hoffschild (1999) [13], Saetra (2004) [14] i Böttner (2007) [15] donen tres exemples sobre com les previsions de l'estat de la mar poden ajudar a la optimització de rutes.

1.2.3 Solucions actuals

Actualment el rutatge a seguir per una nau encara es decideix exclusivament per la capitania de la mateixa. Mitjançant una informació prèvia i la seva pròpia intuïció, la capitania decideix de forma imprecisa quina es al solució a seguir.

No obstant aquesta forma de predir la ruta a seguir, si bé efectiva a termes pràctics, no es gaire aconsellable avui en dia per la disponibilitat, com hem mencionat abans, de mètodes informàtics d'alta precisió.

Aquests mètodes fins ara s'han limitat tan sols a poc més que predir la meteorología de la ruta preestablerta entre dos ports degut a la dificultat inherent del problema a resoldre. El motiu d'aquesta dificultat és que normalment la ruta interessant entre dos punts navegables no es tan sols la línia recta ortodròmica, sinó que s'intenta optimitzar també aspectes com l'estalvi de combustible, el mínim forçament d'un vaixell d'alta envergadura viatjant per baix període d'onatge, la evasió de grans altures d'onatge per millorar el confort, etc. Per aquest motiu, el problema passa

a optimitzar diversos paràmetres, i per tant a ser NP-Difícil i computacionalment costós, fet molt problemàtic degut a que el temps de calcul esperat per aquest tipus de programes es molt baix (degut a la falta de apreciació d'aquesta dificultat del problema per la capitania habitual d'una nau).

Per exemple, a continuació es mostra una imatge del predictor d'onatge de Puertos del Estado (la organització oficial dedicada a la coordinació i eficiència dels ports d'Espanya) per a la ruta de Gijón-Nantes:

Figura 1.1: Predictor d'onatge de Puertos del estado, ruta Gijón-Nantes

Un altre exemple es el de WERMED, que es basa en tindre un nombre molt reduit d'alternatives entre un port i un altre, i escollir entre aquestes rutes prefixades possibles mitjançant la previsió meteorológica en cadascuna d'elles.

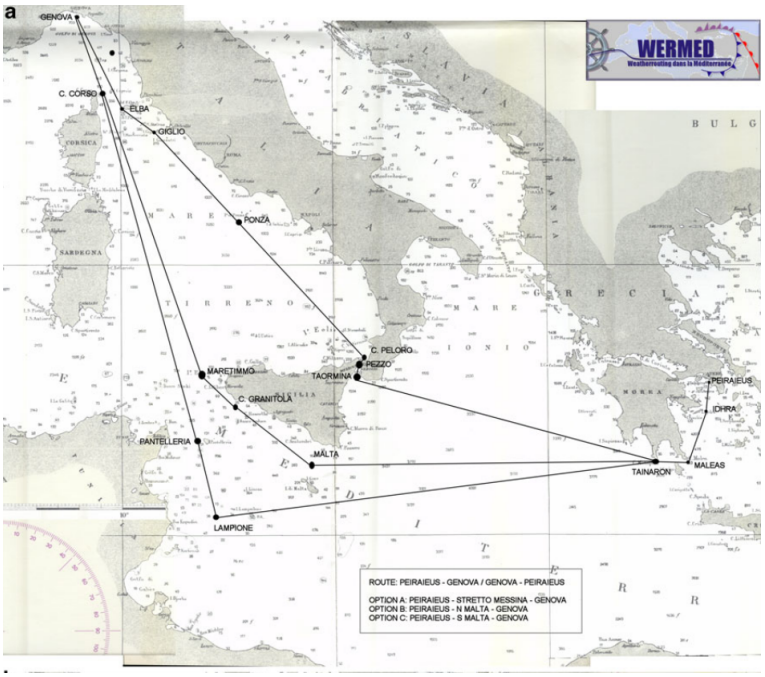


Figura 1.2: Esquema del predictor WERMED per a les rutes entre Génova i Peireos

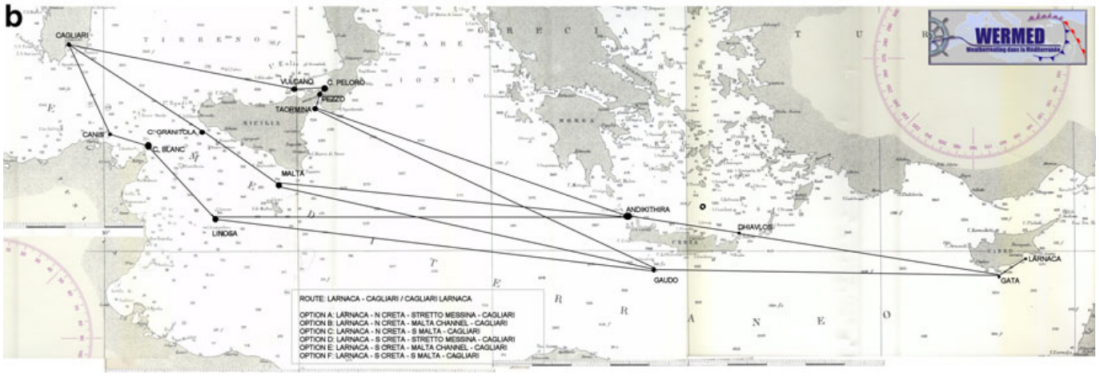


Figura 1.3: Esquema del predictor WERMED per a les rutes entre Cagliari i Larnaca

Capítol 2

Abast

L'abast d'aquest projecte es basa en dissenyar un programa que pugui solucionar el problema de rutatge marítim d'una forma realista com per a poder aplicar les solucions a situacions reals de manera efectiva, i lo suficientment ràpida com per que aquestes solucions es puguin aplicar en temps real. Per fer-ho es proposa una nova perspectiva al problema, adaptant algoritmes de pathfinding multi-objectiu i disposant de dades actualitzades i realistes de l'estat de la mar i de la meteorologia per a crear el programa a realitzar.

2.1 Metodologia i organització

2.1.1 Metodologia

La metodologia triada, si bé senzilla, ha resultat ser molt efectiva per al desenvolupament del projecte. Com era un projecte on l' equip de desenvolupament consistia en una persona (en aquest cas l'estudiant realitzant el projecte), es va decidir seguir una metodologia similar a *Scrum*, que es una metodologia àgil pensada sobretot per al desenvolupament de Software.

Per tant, es va decidir realitzar les reunions pròpies amb el director del projecte de forma setmanal, amb l'objectiu de facilitar la proposta de solucions i la obtenció de feedback. De forma ocasional, també es va decidir realitzar reunions amb els experts de SIMO per aconseguir que el projecte s'adaptés progressivament als resultats esperats per l'usuari final.

Els rols es van assignar de la següent forma:

- Project Master: Es qui s'encarrega de fer seguiment d'allò que realitza l'equip,

en aquest cas el director del treball.

- Product Owner: És qui representa els interessats en el projecte, en aquest cas SIMO o inclús mes endavant Puertos del Estado.
- L'Equip de Desenvolupament: Els qui s'encarreguen de desenvolupar el producte, en aquest cas l'estudiant realitzant el projecte.

Mètode de validació

El mètode de validació proposat és divideix en dos seccions:

1. Per comprovar la correctesa i optimalitat del algoritme, es realitzaran proves de dos tipus:
   (a) Periòdicament es posarà a prova l'algoritme mitjançant un conjunt de jocs de prova preestablerts, que comprendran des de casos senzills fins a casos extrems (depenent dels canvis recents en l'algoritme es pot augmentar o disminuir la freqüència d'aquestes comprovacions).
   (b) A més de les anteriors comprovacions, es sotmetrà l'algoritme a comprovacions amb jocs de prova aleatoris per tal d'assegurar-se de la correctesa general del mateix.
2. Per comprovar la correcta implementació del visualitzador avançat es realitzaran proves a nivell d'usuari (beta-testing) per assegurar-se de que l'aplicació es suficientment intuïtiva i del seu funcionament correcte en tots els àmbits. Òbviament aquestes comprovacions es realitzaran en fases avançades del projecte, quant ja es tingui una versió beta del visualitzador avançat.

2.1.2 Algoritmes i programari

Per a la realització d'aquest projecte sabíem que les alternatives que ens podíem plantejar tindrien que cobrir dos elements diferents: l'algoritme intern de pathfinding que utilitzi el programa i les llibreries de visualització per mostrar el resultat.

- Algoritmes de pathfinding: L'algoritme que utilitzarem per el problema base mono-objectiu pot ser qualsevol algoritme de pathfinding bàsic. En concret, utilitzarem A*, una generalització molt coneguda de Dijkstra [7], ja que ens permet obrir el mínim número possible de nodes per tal de trobar el resultat desitjat.

  Per altra banda, l'algoritme necessari per resoldre el problema multi-objectiu d'aquest projecte ha de ser un algoritme de pathfinding adaptat a aquest tipus de problemes més sofisticats; per tant, les opcions son força limitades. Afortunadament l'algoritme que utilitzarem en aquest projecte, NAMOA*, s'adapta força bé al problema ja d'entrada, i a més es un algoritme que de forma similar al A* pel cas mono-objectiu, es admissible i eficient (tal com descrivim en un apartat posterior).

- Llibreries de visualització: La visualització dels resultats del algoritme es va considerar en un principi secundaria. Si bé es cert que aquesta visualització es essencial per a un suposat usuari final, es va prioritzar la correctesa dels resultats obtinguts per sobre d'una visualització i interacció sobre el programa. Es per aixo que es van triar les llibreries QT, per tal de implementar un senzill visualitzador que pogués mostrar els resultats de forma simple. No obstant, com hem mencionat abans, recentment s'ha decidit implementar, junt amb la col·laboració de SIMO, un visualitzador específic per al programa que mostri els resultats i les dades de forma molt més gràfica.

Capítol 3

Planificació temporal

3.1 Descripció de les tasques

Inicialment, el projecte es va planificar fins al torn de presentació de gener. No obstant, per raons de coordinació amb el director del projecte, es va prorrogar la presentació fins al torn d'abril. Per tant, la planificació del projecte s'ha vist estesa de la següent forma:

- Fase 1 (Fase inicial): Des del 19/05/14 al 20/06/14 al diagrama de Gantt. Fase d'entrar en contacte amb l'entorn i valorar les diferents opcions i algoritmes a implementar.

- Fase 2 (Fase d'implementació bàsica): Des del 23/06/14 al 30/07/14 al diagrama de Gantt. Fase on, un cop decidits els algoritmes, les llibreries de visualització i altres detalls, es comença a implementar un algoritme de path-finding bàsic que realitzi càlculs sobre unes dades mínimament realistes en un entorn de visualització senzill.

- Fase 3 (Fase d'implementació avançada): Des del 31/07/14 al 10/11/14 al diagrama de Gantt. Fase on, una vegada tenim la base, podem realitzar el procediment per acabar implementant el algoritme definitiu del projecte.

- Fase 4 (Fase d'adaptació): Des del 11/11/14 al 02/01/15. Fase on, ja desenvolupat el algoritme, l'adaptem per a que sigui mes especific i realista de cara als aspectes de la navegació marítima.

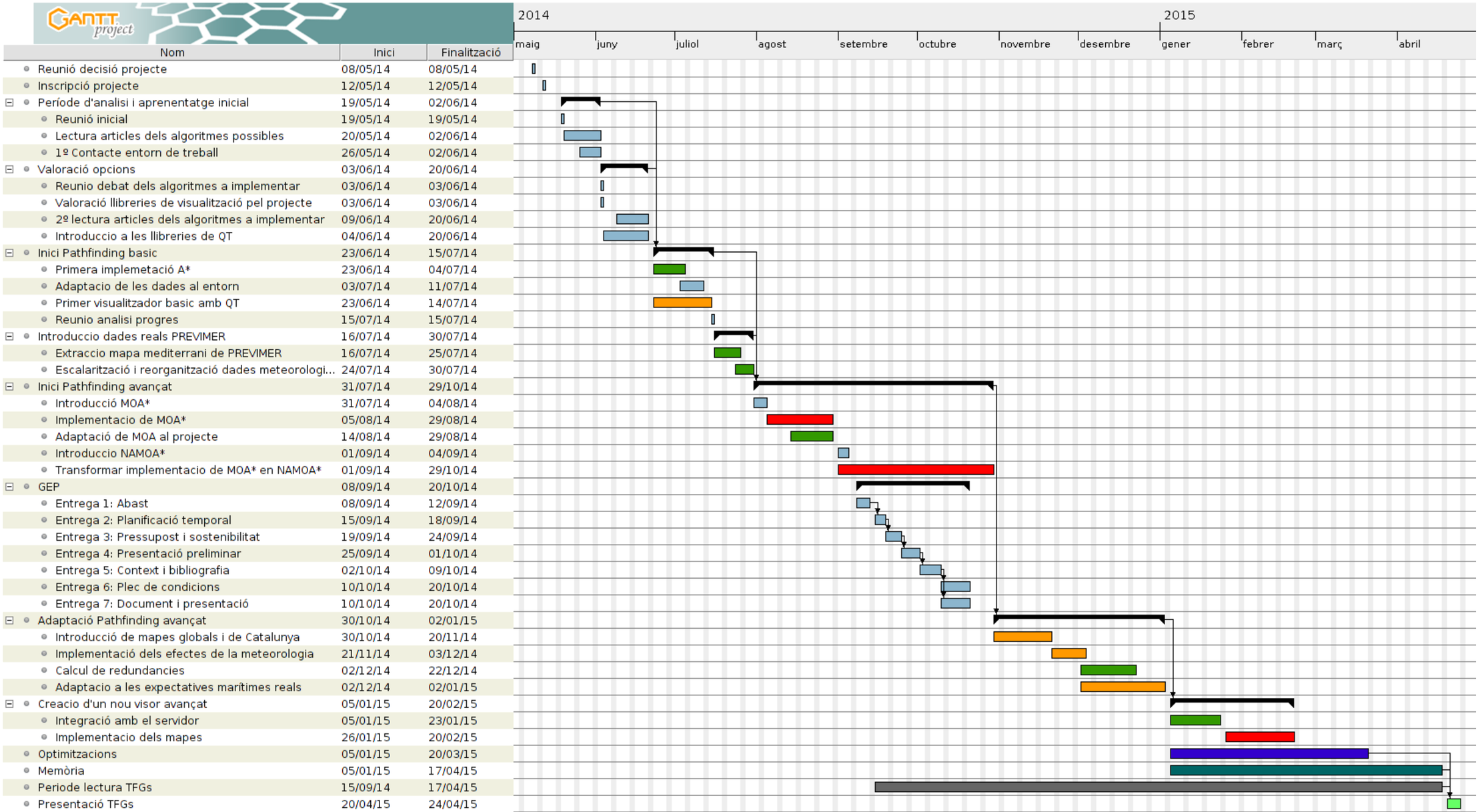
- Fase 5 (Fase de optimitzacions i millores): Des del 05/01/15 al 17/04/15 al diagrama de Gantt. Acabat el programa principal, en aquesta fase es realitza

un visor on es pugui visualitzar de forma molt més clara els resultats. A més, es busquen tot tipus d' optimitzacions per millorar el temps i els resultats del programa.

3.2 Diagrama de Gantt

Aquí podem veure el diagrama de gantt. Les tasques estan classificades per colors de la següent manera:

- Verd: Tasca de risc baix.
- Groc: Tasca de risc mitjà.
- Vermell: Tasca de risc alt.
- Resta de colors: Altres tasques sense risc directe (GEP, memòria, optimitzacions, etc).



Nom	Inici	Finalització
Reunió decisió projecte	08/05/14	08/05/14
Inscripció projecte	12/05/14	12/05/14
Període d'analisi i aprenentatge inicial	19/05/14	02/06/14
Reunió inicial	19/05/14	19/05/14
Lectura articles dels algoritmes possibles	20/05/14	02/06/14
1º Contacte entorn de treball	26/05/14	02/06/14
Valoració opcions	03/06/14	20/06/14
Reunio debat dels algoritmes a implementar	03/06/14	03/06/14
Valoració llibreries de visualització pel projecte	03/06/14	03/06/14
2º lectura articles dels algoritmes a implementar	09/06/14	20/06/14
Introduccio a les llibreries de QT	04/06/14	20/06/14
Inici Pathfinding basic	23/06/14	15/07/14
Primera implemetació A*	23/06/14	04/07/14
Adaptacio de les dades al entorn	03/07/14	11/07/14
Primer visualitzador basic amb QT	23/06/14	14/07/14
Reunio analisi progres	15/07/14	15/07/14
Introduccio dades reals PREVIMER	16/07/14	30/07/14
Extraccio mapa mediterrani de PREVIMER	16/07/14	25/07/14
Escalarització i reorganització dades meteorologi...	24/07/14	30/07/14
Inici Pathfinding avançat	31/07/14	29/10/14
Introducció MOA*	31/07/14	04/08/14
Implementacio de MOA*	05/08/14	29/08/14
Adaptació de MOA al projecte	14/08/14	29/08/14
Introduccio NAMOA*	01/09/14	04/09/14
Transformar implementacio de MOA* en NAMOA*	01/09/14	29/10/14
GEP	08/09/14	20/10/14
Entrega 1: Abast	08/09/14	12/09/14
Entrega 2: Planificació temporal	15/09/14	18/09/14
Entrega 3: Pressupost i sostenibilitat	19/09/14	24/09/14
Entrega 4: Presentació preliminar	25/09/14	01/10/14
Entrega 5: Context i bibliografia	02/10/14	09/10/14
Entrega 6: Plec de condicions	10/10/14	20/10/14
Entrega 7: Document i presentació	10/10/14	20/10/14
Adaptació Pathfinding avançat	30/10/14	02/01/15
Introducció de mapes globals i de Catalunya	30/10/14	20/11/14
Implementació dels efectes de la meteorologia	21/11/14	03/12/14
Calcul de redundancies	02/12/14	22/12/14
Adaptacio a les expectatives marítimes reals	02/12/14	02/01/15
Creacio d'un nou visor avançat	05/01/15	20/02/15
Integració amb el servidor	05/01/15	23/01/15
Implementacio dels mapes	26/01/15	20/02/15
Optimitzacions	05/01/15	20/03/15
Memòria	05/01/15	17/04/15
Periode lectura TFGs	15/09/14	17/04/15
Presentació TFGs	20/04/15	24/04/15

Capítol 4

Gestió econòmica i pressupost

4.1 Identificació dels costos

Podem resumir els costos involucrats en aquest projecte com a tres: recursos humans, de hardware i de software.

4.2 Recursos humans

Per al desenvolupament d'aquest projecte es necessita almenys un/a programador/a que estigui especialitzat en computer science i tingui per tant coneixement previ de les tècniques de pathfinding requerides.

El sou brut anual d'un perfil d'aquest estil, amb titulació superior en enginyeria informàtica, es d'aproximadament uns 32.000 €. Sense tindre en compte les pagues dobles, podem calcular a partir d'aquesta xifra uns 2650 euros aproximadament.

Per calcular els costos reals que suposa per part del contractant tenim que afegir les cotitzacions socials del 29,70% (23,60% de contingències comuns, 5,50% d'atur, 0,60% de formació professional).

Costos per al contractant	**Preus**
Sou brut mensual	2.650 €
+23,60% Cotitzacions contingències comuns	625,4 €
+5,50% Cotitzacions d'atur	145,75 €
+0,60% Cotitzacions de formació professional	15,9 €
Total	3.437,05 €

Taula 4.1: Costos que suposa el treballador per al contractant

Pel que fa al/a la treballador/a se li descompta del sou brut mensual: 6,35% de

cotitzacions socials (4.70% de contingències, 1.55% d'atur, 0,10% de formació professional) i el 20% de IRPF

Sou resultant per al treballador	**Preus**
Sou brut mensual	2.650 €
-4,70% Cotitzacions contingències comuns	-124,55 €
-1,55% Cotitzacions d'atur	-41,08 €
-0,10% Cotitzacions de formació professional	-2,65 €
-20,00% IRPF	-530 €
Total	1951,72 €

Taula 4.2: Sou net que rep el treballador

4.3 Recursos Hardware

Els recursos hardware necessaris per aquest projecte son molt limitats. Bàsicament es necessari un ordinador d'alta gama per a la realització de les proves del algoritme, un ordinador senzill per al control de versions extern, i un servidor per a l'execució externa del programa.

No obstant, Ja es disposava amb anterioritat del ordinador d'alta gama amb anterioritat, i SIMO s'ha oferit a compartir part del seu servidor per a la execució del programa de forma temporal. Per tant, els costos es resumeixen de la següent manera:

Hardware	**Preus**
Ordinador d'alta gama	(1245,00) 0,00 €
Asus Xtion Pro Live	285,00 €
Servidor extern	(60) 0,00 €/mes
Total	285,00 €

Taula 4.3: Preu del hardware

4.4 Recursos Software

Tots els recursos de software que s'utilitzaran no comporten cap despesa. S'ha de tindre en compte que les dades meteorològiques PREVIMER que s'utilitzaran en el

projecte no tenen cap cost degut a que son proporcionades per SIMO

Software	Preus
LLibreries NetCDF	0,00 €
Dades meteorològiques PREVIMER	0,00 €
Sistema operatiu Ubuntu 14.04.1 LTS	0,00 €
Total	0,00 €

Taula 4.4: Preu del software

4.5 Resum costos

Per tant, podem definir els diferents pressupostos de recursos humans, de hardware i de software de la següent forma:

Concepte	Preus
Salaris	3.437,05 €
Amortitzacions hardware	44,53 €
Manteniment llicencies software	0,00 €
Total	3.481,58 €

Taula 4.5: Resum costos mensuals

El cost total d'aquest projecte, si tenim en compte que es realitza durant 11 mesos, es descriuria de la següent forma:

Concepte	Preus
Salaris	37.807,55 €
Amortitzacions hardware	285,00 €
Manteniment llicencies software	0,00 €
Total	38092,55 €

Taula 4.6: Resum costos totals

S'ha de tindre en compte que el cost real d'aquest projecte, al estar realitzat dintre del àmbit d'un Treball Final de Grau de modalitat A, seria nul en els salaris. A més, es probable que la duració es pogués veure reduïda tenint en compte que no caldria la realització de GEP.

Capítol 5

Sostenibilitat i compromís social

5.1 Valoració de la sostenibilitat del projecte

La viabilitat econòmica d'aquest projecte es força alta, principalment perquè els recursos invertits son mínims, i en el cas de que el projecte acabi sent utilitzat per alguna empresa, com es probable ja que hi han diverses d'interessades (SIMO, Puertos del Estado, etc), el retorn d'inversió resultant serà elevat. A més, els costos de material son mínims, ja que es un projecte purament de programació, i la majoria del material utilitzat ja havia sigut adquirit anteriorment. Per aquest mateix motiu, pel fet de reutilitzar material existent, la valoració ambiental es molt alta. A més, una vegada s'acabi la vida útil del material d'aquest projecte es pot seguir reutilitzant donant-lo, per exemple, a la ONG de la UPC Tecnologia per a tothom (TXT), que permet reutilitzar ordinadors antics per donar-los a gent mes necessitada. Finalment, la viabilitat social d'aquest projecte es també bona, ja que amb la seva aplicació es podrà millorar l'eficiència dels viatges nàutics, i per tant, suposarà una millora tant per la tripulació del vaixell com pels costos del viatge de cara als usuaris.

En els següents apartats es detallen els aspectes econòmics, socials i ambientals responent a una serie de preguntes sobre els mateixos.

5.2 Aspectes econòmics

- *El cost del projecte ho faria viable si hagués de ser competitiu?*

Si, ja que com a recursos només necessita d'un sol especialista en computer science, i el hardware i software es molt limitat o directament gratuït (de hardware tan sols es imprescindible un ordinador, i de software tan sols les llibreries Qt i NetCDF, les bases de dades de BingMaps, un compilador de C++, una IDE i òbviament el sistema operatiu Ubuntu. No obstant si es vol pujar el programa a un servidor seria necessari el manteniment del mateix). A part d'aquestos elements, s'hauria de tindre cura de que es pogués comptar amb la col·laboració d'alguna empresa del sector, com SIMO, que tingués accés a les dades marítimes necessàries per al projecte.

- *Es podria realitzar un projecte similar en molt menys temps o amb molts menys recursos, i per tant menor cost?*

Es segur que amb més recursos, com per exemple més programadors es podria realitzar aquest projecte més ràpidament. No obstant es difícilment possible que es pogués obtindre un menor cost en general ja que, com hem dit a la pregunta anterior, els recursos son mínims.

- *El temps dedicat a cada tasca és proporcional a la seva importància (s'ha dedicat molt de temps a desenvolupar parts del projecte que podien haver estat reutilitzades de tecnologies / projectes / coneixements existents)?*

Totes les tasques desenvolupades en aquest projecte parteixen d'elements ja existents, ja sigui l'algoritme (el A* es un dels algoritmes més coneguts de computer science, i del NAMOA* es disposa dels articles que el descriuen a fons) les dades marítimes proporcionades per SIMO a partir de les de PREVIMER, o el visor final que utilitza el WMS de Bing Maps.

5.3 Aspectes socials

- *Hi ha una necessitat real del teu producte / servei?*

Els programes suport a la navegació marítima d'avui en dia son molt limitats i específics com hem vist a l'apartat de l'estat de l'art, normalment es basen en rutes limitades ja conegudes i en prediccions sobre l'estat del mar sobre les mateixes. Amb aquest projecte es pretén que les rutes siguin calculades per l'algoritme depenent de l'estat actual del mar i la meteorologia i de les preferències del capità del vaixell, cosa que permetrà obtindre una ruta que s'adapti a la situació actual i ajudi al capità de forma molt més eficient.

- *Satisfer aquesta necessitat, millora la qualitat de vida dels consumidors?*

Òbviament rutes mes òptimes i eficients signifiquen preus mes baixos per als consumidors.

5.4 Aspectes ambientals

- *Quins recursos es poden reaprofitar d'altres projectes?*

La majoria del hardware que s'utilitzarà en aquest projecte no s'ha tingut que adquirir per a la realització del mateix.

- *S'ha tingut en compte el desmantellament una vegada acabi la vida útil del TFG? Com afectarà la desmantellament del procés / producte al medi ambient? Si és un producte, s'han tingut en compte en el seu disseny criteris de facilitació de les seves posterior reciclatge?*

Una vegada acabi la vida útil dels ordinadors utilitzats en aquest projecte es poden donar a la associació per a la reutilització de ordinadors de la UPC anomenada TxT (Tecnologia per Tothom) que readaptaran els ordinadors per poder distribuir-los a persones més necessitades.

Part III

Definició i Implementació

Capítol 6

Descripció detallada del problema

6.1 Dades del problema

- **La malla.** Tenim una malla de dimensió $n \times m$. Ens referim als punts de la malla com x_{ij}. En general, cada punt x_{ij} serà *adjacent* als seus 8 veïns + 32 veïns addicionals arbitraris (introduïts per millorar el número de rumbs possibles, veure la imatge a continuació per a una descripció gràfica). Els punts de la malla que corresponguin amb *terra* no seran veïns de ningú. Ens referirem als veïns d'un punt x_{ij} com N_{ij}.

- **El temps.** El viatge comença en un instant de temps ts i s'incrementa a partir del mateix depenent de la meteorología que hi hagi en els instants de temps t_k, t_{k+1} entre cada parell de nodes recorreguts x_{ij} i de la velocitat utilitzada v_l

- **La meteorología.** Disposem de k funcions $m_i(u, t)$ que retornen la predicció meteorològica en el punt u en el temps t. Cada i es refereix a un paràmetre com l'*altura d'ona*, la *direcció de les corrents*, el *període del onatge*, etc.

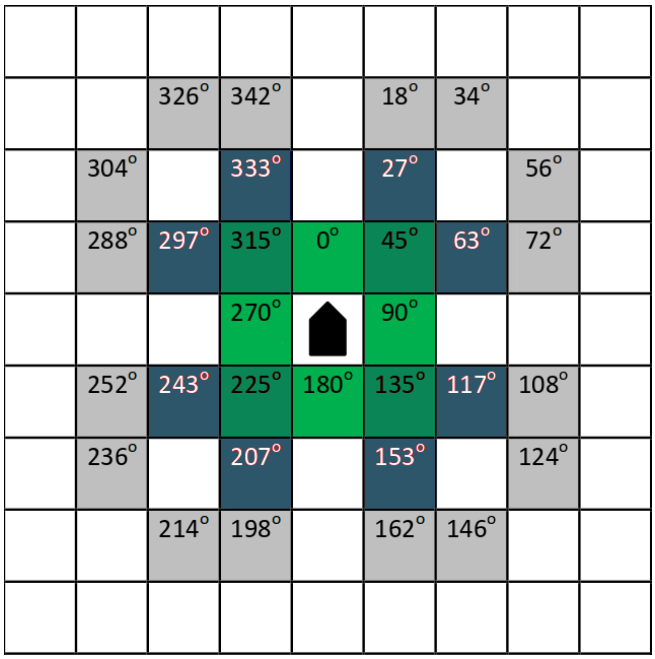


Figura 6.1: Possibles rumbs del vaixell

6.2 Definició de ruta

Donat un punt d'origen x_{ab} i un punt de destí $x_{a'b'}$, una ruta es una seqüencia de parells: $(u_0, t_0), (u_1, t_1), ..., (u_n, t_n)$ tals que:

- $u_0 = x_{ab}, t_0 = ts$
- $u_n = x_{a'b'}, t_n = ts + \sum_{i=0}^{n-1} timecost(u_i, u_{i+1}, t_i, v_l)$
- u_i es veí de u_{i+1} per a tot $i = 0..n-1$

Observació: La funció timecost, utilitzada per calcular el cost temporal de viatjar d'un node a un altre, esta descrita en detall en el següent apartat, ja que també s'utilitzarà com una de les funcions heurístiques.

6.3 Qualitat d'una ruta respecte a un criteri

La definició del problema ens condueix a establir diferents funcions heurístiques per avaluar la qualitat d'una ruta. Suposem que una ruta R $= (u_0, t_0), (u_1, t_1), ..., (u_n, t_n)$, llavors definim les funcions heurístiques de la següent manera:

- *Temps.*

$$\sum_{i=0}^{n-1} timecost(u_i, u_{i+1}, t_i, v_l)$$

on $timecost(u_i, u_{i+1}, t_i, v_l)$ es una funció que retorna el cost temporal de viatjar del punt u al seu veí u' tenint en compte les prediccions meteorològiques entre tots dos punts desde el instant de temps t_i utilitzant la velocitat v_l.

- *Confort.*

$$\max_{i=0}^{n-1} confort(u_i, u_{i+1}, t_i, t_{i+1})$$

on $confort(u_i, u_{i+1}, v_i)$ es una funció que retorna la altura d'ona màxima trobada al llarg del recorregut. Aquest heurístic es un màxim degut a que lo important en aquest aspecte generalment es lo dolenta que ha sigut la pitjor etapa.

6.4 Número de paràmetres i definició de ruta òptima

El nivell de dificultat del problema depèn en gran mesura de la quantitat de paràmetres a optimitzar. Si optimitzem un paràmetre el problema serà resoluble per un algoritme de pathfinding bàsic, com l'A*, no obstant quant es necessari optimitzar mes d'un paràmetre es necessita un algoritme de pathfinding multi-objectiu.
De la mateixa manera, la definició de la ruta òptima també es veurà afectada de la següent manera:

- **Un paràmetre, optimalitat:** Es considera òptima la ruta R si i sol si no existeix un altra ruta R' tal que $f(R') \geq f(R)$

- **Més d'un paràmetre, pareto-optimalitat:** Es consideren òptimes totes les rutes R que no siguin pareto-dominades, es a dir, si i sol si no existeix un altra ruta R' que per tot i, $f_i(R') \geq f_i(R)$

Capítol 7

Descripció dels algoritmes base

En aquesta secció veurem el funcionalment dels algoritmes principals en els que es basa aquest projecte, per tal d'introduir les modificacions que realitzarem posteriorment per a la resolució del problema i de definir aspectes clau de la mateixa.

7.1 Algoritme A*

7.1.1 Definició del problema de cerca mono-objectiva

Donat un graf dirigit finit amb pesos no negatius $G = (N, A, c)$, consistent de $|N|$ nodes i $|A|$ arestes (dirigides) amb costos (positius) $c(n, n) \in \mathbb{R}$.

Donat un node inicial s i un node final t que son accessibles entre si (dos nodes n_i n_j son accessibles si i sol si existeix un camí entre n_s i n_t).

Definirem el camí entre dos nodes qualsevol n_s i n_t com una serie de nodes $(n_1, n_2, ..., n_k)$ en els que cada n_{i+1} es successor de n_i (un node n_{i+1} es successor de n_i si i sol si existeix una aresta que arriba a n_{i+1} des de n_i). Cada camí té un cost associat $cost(n_s, n_t)$, que es el cost que obtenim al sumar cadascun dels costos pertanyents a cadascuna de les arestes $m_{i,i+1}$ en el camí.

Definirem el camí òptim entre dos nodes qualsevol n_i i n_j com el camí amb un cost associat mínim entre tots els camins possibles entre els dos nodes, es a dir, $min(cost(n_i, n_j))$.

Donat un node d'inici $s \in N$, i un conjunt de nodes destí $\Gamma \subseteq N$, l'objectiu es trobar el camí òptim en G de s a un node destinació pertanyent a Γ.

7.1.2 Pseudocodi

Assumim que es disposa d'una funció heurística $h(n)$ que per a un node n qualsevol que indica el cost fins a arribar al node t destinació de forma que $h(n) <= min(cost(n_i, n_j))$ (es a dir, menor o igual al **camí òptim**). Assumim també que calcularem la funció d'avaluació $f(n) = g(n) + h(n)$ que utilitzarem per avaluar quin es el millor node per expandir a cada pas, on g(n) es el cost per arribar desde el node inicial s fins a n.

1. CREAR:
   - $G_{op} = s$: El conjunt de possibles nodes a ser avaluats, inicialment conté el node inicial.
   - $G_{cl} = \emptyset$: El conjunt de nodes ja avaluats.
2. SELECCIÓ DEL CAMÍ. Seleccionar el node n de G_{op} tal que n sigui el node amb un valor mínim (o màxim) en la funció d'avaluació $f(n)$, es a dir:

   $$\forall n \in G_{op} \quad \nexists n' \in G_{op} \quad | \quad f(n') < f(n)$$

   S'ha de notar que els desempats de nodes amb el mateix valor de la funció d'avaluació es poden resoldre arbitràriament, però sempre a favor del node t.
3. COMPROVAR FINALITZACIÓ: Si el node t es el node seleccionat finalitzem l'algoritme
4. EXPANSIÓ DEL CAMÍ: Si el node n seleccionat no es el node destinació, l'eliminem de G_{op} i l'afegim a G_{cl}. Llavors afegim a G_{op} tots els nodes successors de n i retornem al pas de selecció de camí.

Com es pot intuir, l'algoritme A* es basa en una selecció de camins correctament informada per l'heurístic per obrir tan sols els nodes necessaris per arribar a la solució òptima. De fet, donat un heurístic amb les propietats que descriurem a continuació, l'algoritme A* serà admissible, òptim i eficient.

7.1.3 Admissibilitat

Assumint que $h*(n)$ indica el cost real del camí des de un node n fins al node t destinació. Quant l'heurístic es una estimació optimista ($h(n) \leq h*(n) \forall n$), la cerca es **admissible**, es a dir, si una solució existeix es garantit que en trobara la solució òptima [8].

7.1.4 Eficiència

L'eficiència del A* està relacionada estretament amb la exactitud de la funció heurística $h(n)$. En concret, resultats importants poden ser provats quant $h(n)$ es consistent o satisfà la propietat d'equivalència monòtona. Assumint que la funció $k(n, n')$ indica el cost del camí optim des de n a n' en el graf G, una funció heurística $h(n)$ es consistent quant,

$$h(n) \leq k(n, n') + h(n') \quad \forall (n, n') \in N$$

Alternativament, $h(n)$ es una funció monòtona quant,

$$h(n) \leq c(n, n') + h(n') \quad \forall (n, n') \in A$$

De fet, quant $h(n) = 0$ per tots els nodes, A* es comportarà com l'algoritme de Dijkstra.

Suposant una funció $h(n)$ consistent A* soluciona el problema en cas pitjor en $O(|N|)$ iteracions, emmagatzemant $O(|N|)$ nodes en memòria. En particular, si $c*$ denota el cost de la solució òptima, A* sempre expandira els nodes en els que $f(n) < c*$, i com a màxim els nodes en els que $f(n) \leq c*$. Per tant, donat una estimació heurística consistent, valors més grans de $h(n)$ poden empentar l'avaluació de mes i mes nodes més enllà de la frontera $f(n) = c*$, reduint l'esforç de la cerca.

7.1.5 Optimalitat

Quant l'heurístic es consistent, A* també es pot provar com a òptim entre la classe dels algoritmes admissibles tant en termes dels nodes expandits com en el número

de expansions de nodes (iteracions) [16]. Això vol dir que no hi ha cap algoritme d'aquesta classe que pugui evitar una expansió d'un node realitzada per l'A* sense comprometre l'admissibilitat.

7.2 Algoritme NAMOA*

7.2.1 Definició del problema de cerca multi-objectiva

Donat un graf dirigit finit amb pesos no negatius $G = (N, A, \vec{c})$, consistent de $|N|$ nodes i $|A|$ arestes (dirigides) amb vectors de costos (positius) $\vec{c}(n, n') \in \mathbb{R}^q$.
Definim un camí en G com la seqüencia de nodes $P = (n_1, n_2, ...n_k)$ tal que per cada $i \leq k$, $(n_i, n_{i+1}) \in A$, i el vector de costos associat a cada camí $\vec{c}(P)$ sigui la suma dels vectors de costos dels arcs que el componen.
Donat un node d'inici $s \in N$, i un conjunt de nodes destí $\Gamma \subseteq N$, l'objectiu es trobar el conjunt de camins **no dominats** entre si en G de s als nodes destinació Γ.

7.2.2 Definició de dominància

En els problemes multi-objectiu, dels vectors de costos $\vec{c}(n, n')$ se'n deriva tan sols una relació parcial de preferència d'ordre $\prec$ anomenada *dominància*.

$$\forall \vec{f}, \vec{f'} \in \mathbb{R}^q \quad \vec{f} \prec \vec{f'} \quad \Leftrightarrow \quad \forall i \quad f_i \leq f_{i'} \wedge \vec{f} \neq \vec{f'}$$

On f_i es refereix al i-èsim element del vector $\vec{f}$. Per tant, donats dos vectors q-dimensionals $\vec{f}$ i $\vec{f'}$, no sempre es possible avaluar un com a millor que l'altre. Per exemple, en un espai de costos bidimensional, el vector $(2, 3)$ domina al vector $(2, 4)$, però no existeix cap relació de dominància entre els vectors $(2, 3)$ i $(3, 2)$. Per tant, donat un conjunt de vectors X, definirem com $nondom(X)$ el conjunt de vectors no dominats entre si en el conjunt X de la següent manera,

$$nondom(X) = \vec{x} \in X | \nexists \vec{y} \in X \quad \vec{y} \prec \vec{x}$$

7.2.3 Pseudocodi

Com podrem veure al següent pseudocodi, el problema de cerca multi-objectiva es pot relatar en gran part a la cerca mono-objectiva. No obstant, s'han de destacar certes diferencies de gran importància: Per començar, degut a la relació de dominància, dos (o més) camins no comparables (no dominats) poden arribar al mateix node pels mateixos o per diferents nodes pare. Es per aixo que es generen diverses diferencies:

- L'arbre de cerca utilitzat per A* per enregistrar el millor camí conegut ja no es suficient. Es necessari un graf acíclic, per tal d'enregistrar el conjunt de camins no dominats entre si als diferents nodes. Per tant, anomenarem a aquesta estructura de dades graf de cerca, o **search graph** (SG).

- El nombre de nodes generats pot ja no ser una estimació realista de la memòria utilitzada per l'algoritme. Un nombre variable de arcs i de vectors de costos no dominats entre si arribant a cada node han de ser enregistrats també.

- Finalment, cada vegada que un nou camí es generat cap a un node conegut, pot ser que el seu cost hagi de ser comparat per veure si hi ha dominància amb el conjunt de tots els costos enregistrats dels camins coneguts que arriben fins al node. El cost computacional d'aquesta operació augmenta amb el número de vectors de cost emmagatzemats a cada node.

Definirem com a $\vec{g}(P)$ el vector de costos de cadascun dels camins emmagatzemats al graf de cerca SG. En general, les estimacions del heurístic correspondran a un conjunt de vectors $H(n)$ per cada node n, estimant vectors de costos des de n fins a cada node destinació. Per tant, per cada camí P_{sn} des de s fins a n amb cost $\vec{P} = \vec{g}_P$, hi hauran un conjunt de vectors d'avaluació heurística, $F(P_{sn})$. Aquesta es la funció anàloga multi-objectiva a $f(n)$ en $A*$,

$$F(P_{sn}) = F(n, \vec{g}_P) = nondom\vec{f} | \vec{f} = \vec{g}_P + \vec{h} \wedge \vec{h} \in H(n)$$

Finalment, s'ha de notar que en cada iteració, A* selecciona i expandeix un node obert, es a dir, considera totes les possibles extensions del camí emmagatzemat en l'arbre de cerca per aquell node. En A*, cada node obert representa una única solució parcial del camí que pot ser expandit mes enllà. En canvi, en el cas multi-objectiu, com que un graf acíclic es utilitzat en comptes d'un arbre per al enregistrament de solucions, aquest ja no es el cas.

1. CREAR:
   - Els conjunts $G_{op}(s) = \vec{0}$, $G_{cl}(s) = \emptyset$.
   - Un graf acíclic SG amb arrel a s
   - Una llista d'alternatives, $OPEN = (s, \vec{g_s}, F(s, \vec{g_s}))$.
   - Dos conjunts buits, $GOALN, COSTS$.
2. COMPROVAR FINALITZACIÓ: Si $OPEN$ esta buit, llavors busca en sentit invers a SG des dels nodes en $GOALN$ tenint en compte els costos en $COSTS$ i retorna el subgraf de la solució.
3. SELECCIÓ DEL CAMÍ. Seleccionar una alternativa $(n, \vec{g_n}, F)$ de $OPEN$ amb $\vec{f} \in F$ no dominat a $OPEN$, es a dir

$$\forall (n, \vec{g_n}, F) \in OPEN \quad \nexists \vec{f'} \in F' \quad | \quad \vec{f'} \prec \vec{f}$$

   Eliminar $(n, \vec{g_n}, F)$ de $OPEN$, i moure $\vec{g_n}$ des de G_{op} a G_{cl}.
4. ENREGISTRAMENT DE LA SOLUCIÓ. Si $n \in \Gamma$, llavors
   - Incloure n a $GOALN$ i $\vec{g_n}$ a $COSTS$.
   - Per a totes les alternatives $(x, \vec{g_x}, F_x)$ en $OPEN$, eliminar de F_x tots els vectors dominats per $\vec{g_n}$.
   - Eliminar de $OPEN$ totes les alternatives $(x, \vec{g_x}, F_x)$ tals que F_x es buida.
   - Retornar al pas de *COMPROVAR FINALITZACIÓ*.

5. EXPANSIÓ DEL CAMÍ. Si $n \notin \Gamma$, llavors: Per a tots els nodes successors m de n que no produeixin cicles a SG:

   - Calcular el cost del nou camí trobat a m: $\vec{g_m} = \vec{g_n} + \vec{c}(n, m)$.
   - Si m es un nou node:
     - Calcular $F_m = F(m, \vec{g_m})$ filtrant estimacions dominades a $COSTS$.
     - Si F_m no esta buit, insertar $(m, \vec{g}_m, F_m)$ a $OPEN$, establir $G_{op}(m) = \vec{g_m}$, i etiquetar $\vec{g_m}$ amb un punter a n.
   - Sino, si $\vec{g_m}$ es igual a algun vector de costos en $G_{op}(m) \cup G_{cl}(m)$ llavors
     - Etiquetem $\vec{g_m}$ a un apuntador cap a n.
   - Sino, si $\vec{g_m}$ no esta dominat per cap vector en $G_{op}(m) \cup G_{cl}(m)$ (un camí cap a m amb un nou cost interessant ha sigut trobat), llavors:
     - Eliminar de $G_{op}(m)$ i $G_{cl}(m)$ vectors dominats per $\vec{g}_m$ (poda). Per cada vector $\vec{g'}$ eliminat de $G_{op}(m)$, eliminar la tupla $(m, \vec{g'}, F)$ de $OPEN$.
     - Calcular $F_m = F(m, \vec{g}_m)$ filtrant estimacions dominades per $COSTS$.
     - Si F_m no esta buit, posar $(m, \vec{g}_m, F_m)$ en $OPEN$, i posar $\vec{g}_m$ en $G_op(m)$ etiquetant un apuntador cap a n.

   Fi del FOR. Retornar al pas de *COMPROVAR FINALITZACIÓ*.

7.2.4 Admissibilitat

En el cas de la cerca mono-objectiva, afirmem que un algoritme es *admissible* si està garantit que el algoritme retornarà la solució òptima sempre que una solució existeixi. Podem extendre aquesta afirmació al cas de la cerca multi-objectiva de la següent manera: un algoritme multi-objectiu es *admissible* si finalitza amb el conjunt de solucions no dominades entre si quant aquest conjunt es finit i no buit, o si no finalitza quant aquest conjunt es infinit [17]. En el cas de NAMOA*, l'algoritme finalitza tan sols quant el conjunt $OPEN$ es buida, i això no pot passar excepte

en el cas de que s'hagin trobat **totes** les solucions no dominades [1][18], i per tant, suposant que el conjunt d'heurístics $H(n)$ siguin optimistes, NAMOA* es admissible.

7.2.5 Eficiència

L'anàlisi d'eficiència de NAMOA* paralelitza el de l'A*, tot i que hi ha certes diferencies degut a la naturalesa del problema de cerca multi-objectiva, com sobretot que l'algoritme ha de cercar totes les solucions no dominades. En aquest cas la eficiència de NAMOA* també es basa principalment en lo informat que estigui la funció heurística multi-objectiva $H(n)$ que utilitzi, i la seva eficiència pot ser provada quant $H(n)$ es **consistent**. Una funció heurística multi-objectiva es consistent quant per a tots els parells de nodes n, n' en el graf, per a tots els camins no dominats entre els dos nodes $P = (n, ..., n')$, i per tots els vectors de cost heurístics $\vec{h'} \in H(n')$, la següent condició es manté,

$$\forall \vec{h'} \in H(n') \quad \exists \vec{h} \in H(n) \quad | \quad \vec{h} \preceq \vec{c}(P) + \vec{h'}$$

Alternativament, $H(n)$ es una funció monòtona quant la següent condició es manté,

$$\forall \vec{h'} \in H(n') \quad \exists \vec{h} \in H(n) \quad | \quad \vec{h} \preceq \vec{c}(n, n') + \vec{h'}$$

Definim $C*$ com el conjunt de costos de les solucions no dominades, i definim $\vec{c*}$ com un dels membres de $C*$. Suposant una funció heurística multi-objectiva consistent. Suposant C com un conjunt qualsevol de vectors de costos.
Definim que un camí qualsevol $P = (s = n_0, n_1, n_2, ..., n_k)$ es **C-limitat** respecte $H(n)$ si per a tots els subcamins $P_i = (n_0, n_1, ..., n_i)$ de P la següent condició es manté,

$$\exists \vec{h} \in H(n_i) | \nexists \vec{c} \in C, \quad \vec{c} \prec \vec{g}(P_i) + \vec{h}$$

Per tant, si disposem d'una funció heurística multi-objectiva consistent es garanteix que tan sols els camins C*-limitats estrictament necessaris seran escollits per a expandir-se. En altres paraules, la poda dels camins C*-limitats serà òptima.

7.2.6 Optimalitat

La cerca multi-objectiva es coneguda per tindre uns requeriments tant de temps com d' espai exponencials en cas pitjor. No obstant es important veure que parlar del número de nodes expandits ja no es una mesura de l'eficiència en el cas multi-objectiu [17]. Això es degut al fet de que diferents camins no dominats poden arribar a cadascun dels nodes en el graf de cerca (SG). Amb una profunditat elevada, un nombre de camins exponencialment gran poden arribar a cada node en el cas pitjor. De fet, el nombre promig de vectors q-dimensionals no dominats en un conjunt de tamany L és $O((log|L|)^{q-1})$, suposant que tots els $(n!)^q$ ordenacions relatives siguin igualment probables [19]. Per tant, es el nombre de vectors de costos emmagatzemats en el graf de cerca el que domina els requeriments d'espai en el cas general. De fet, anàlogament a A*, NAMOA* mostra les propietats d'optimalitat entre la classe d'algoritmes admissibles multi-objectius tant en nombre de vectors de costos emmagatzemats en el graf com en el nombre de expansions de camins [1].

Capítol 8

Descripció del algoritme

Si bé l'algoritme a implementar es basarà en l'algoritme NAMOA* descrit en l'anterior apartat, la naturalesa del problema fa que s'hagi d'adaptar en alguns aspectes que descriurem a continuació.

8.1 Complexitat del problema

En una primera impressió després de la descripció del problema podria donar la sensació de que el problema a resoldre es un cas molt estàndard de pathfinding multi-objectiu. No obstant, hi han diversos factors a tindre en compte que fan que la complexitat del problema estigui lluny de ser trivial:

- Per començar i potser el més important, s'ha de notar que un problema multi-objectiu, per molt senzill que sigui, no deixa de ser **NP-Complet**, i per tant el cost computacional per a la resolució del problema es molt elevat, sobretot quant el problema es de dimensions molt grans. En aquest cas la matriu a resoldre pot ser de dimensions molt elevades (en el cas del mapa global son 720x317 nodes, es a dir, 228240 nodes en total) i per tant el temps de calcul, que s'espera breu per a una correcta interacció amb l'usuari final, es complexe de complir.

- A més, s'ha de tindre en compte que en el problema a resoldre es té en compte el **temps**, factor que introdueix una nova dimensió al problema, ja que depenent del temps *time* en que ens trobem en un node n, **l'estat de la meteorologia pot ser diferent**, i per tant les possibilitats per arribar al node destinació seran exponencialment més elevades. De fet, aquesta característica es de gran

importancia, i com descriurem més endavant cal adaptar l'algoritme per tal de que només tractem un nombre limitat de casos a favor del temps de calcul, tot i que en detriment de l'admissibilitat.

- Finalment, també disposarem d'un nombre mes gran que 1 de **velocitats diferents**, i per tant caldrà tindre-les en compte de forma que no disparin el cost computacional, ja que potencialment donades V_n velocitats diferents es podria elevar el temps de calcul en n vegades.

8.2 Adaptacions de l'algoritme

8.2.1 Canvis en les estructures de dades

Les estructures de dades utilitzades en l'algoritme del projecte tenen algunes diferencies importants respecte a les de NAMOA*. Si bé alguns d'aquests canvis son deguts simplement al transcurs de la descripció del algoritme a la seva implementació en si (com en les comprovacions de dominància), altres canvis son propis del nou algoritme, generalment per adaptar-se al problema i minimitzar temps de calcul. Els canvis realitzats son els següents:

- Un dels canvis més importants es la eliminació del graf acíclic SG. Aquest canvi principalment es degut la possibilitat d'evitar l'alt cost computacional que requereix la comprovació a SG de que no es produeixin cicles. Les conseqüències de no utilitzar aquesta estructura son negligibles sempre que en a ordenació dels nodes de $OPEN$ es prioritzin certs heurístics (com per exemple distancia, combustible, etc) que evitin els cicles sempre que els costos referits en aquests heurístics mai siguin menors a zero.

- En aquest algoritme, els camins possibles que estem considerant, o que ja estan emmagatzemats al *closedset* d'un node, es refereixen com a *possibilitats*. Aquest canvi es simplement per a facilitar la referenciació independent d'un d'aquests camins possibles, i la seva estructura serà $(n, \vec{g_{ni}}, \vec{f_{ni}})$, on n es el node

al que ens referim, $\vec{g_{ni}}$ el seu vector de costos fins a arribar al node actual i $\vec{f_{ni}}$) el vector d'estimacions dels heurístics més $\vec{g_{ni}}$.

- La estructura $COSTS$ que en NAMOA* es refereix als costos dels camins ja calculats que arriben a la solució, en aquest algoritme es defineix directament com a $Gcl(t)$, ja que sabem que tan sols existeix un node destinació simultàniament al problema (tot i que teòricament equival a $Gcl(t) \cup Gop(t)$ però ho definim així ja que sempre es compleix que $Gop(t) = \emptyset$).

- Una matriu $NODES$ que simplement s'ha afegit per representar de forma explícita el conjunt de $Gop(n)$ i $Gcl(n)$ de tots els nodes.

8.2.2 Comprovacions de dominància

Aquest es un aspecte important a l'hora de implementar un algoritme multi-objectiu, ja que al ser $OPEN$ una cua de prioritats (o en general qualsevol estructura d'ordenació per prioritat) hi haurà moments en els que haguem de filtrar una possibilitat de $OPEN$, i per tant no puguem. Per adreçar aquesta qüestió, comprovem dominància en dos punts:

- Al considerar una nova possibilitat $(n, \vec{g_{ni}}, \vec{f_{ni}})$ en un node n, es comprova dominància respecte $G_{op}(n) \cup G_{cl}(n)$. Òbviament si $(n, \vec{g_{ni}}, \vec{f_{ni}})$ es dominada per alguna possibilitat $(n, \vec{g_{nj}}, \vec{f_{nj}}) \in G_{op}(n) \cup G_{cl}(n)$ llavors simplement es descarta i es segueix la execució del algoritme amb normalitat.

  De la mateixa forma, si $(n, \vec{g_{ni}}, \vec{f_{ni}})$ domina a $(n, \vec{g_{nj}}, \vec{f_{nj}}) \in G_{cl}(n)$, llavors aquesta possibilitat es descarta.

  No obstant, si $(n, \vec{g_{ni}}, \vec{f_{ni}})$ domina a alguna possibilitat $(n, \vec{g_{nj}}, \vec{f_{nj}}) \in G_{op}(n)$, **llavors aquesta no es pot filtrar en aquell instant** degut a que també està dintre de $OPEN$. Per tant, no hi ha més remei que, de sortir aquesta possibilitat no valida de $OPEN$, filtrar-la, però el problema llavors es que *No podem saber quant considerarem aquesta possibilitat (ara nul·la).* Per tant,

hem d'establir una comprovació de dominància addicional per cada possibilitat que considerem des de $OPEN$ per tal de que no sigui una possibilitat nul·la.

- Al traspassar una possibilitat $(n, \vec{g_{ni}}, \vec{f_{ni}})$ des de $G_{op}(n)$ fins a $G_{cl}(n)$, també hem de realitzar una comprovació addicional, ja que es possible que en el temps des de que aquesta possibilitat va ser inserida a $OPEN$ fins a la seva consideració final, hagin canviat els camins que s'han trobat fins aquell moment, es a dir, haurem de comprovar dominància respecte les possibilitats emmagatzemades a $G_{cl}(t)$.

8.3 Pseudocodi

1. INICIALITZAR:
   - El graf $G = (N, A, \vec{c})$ a partir de les dades meteorològiques disponibles.
   - Els nodes s d'inici i t destinació a partir de les dades de latitud i longitud d'entrada.
   - Una cua de prioritats $OPEN = s$ on s'emmagatzemen el conjunt ordenat de possibilitats a ser avaluades
   - Una matriu $NODES$ on es representa cada node junt amb les seves possibilitats a ser avaluades ($G_{op}(N)$) i les possibilitats ja avaluades no dominades entre si en aquell node ($G_{cl}(N)$).
2. COMPROVAR FINALITZACIÓ: Si $OPEN$ esta buit i $G_{cl}(t)$ no esta buit, llavors per cadascuna de les possibilitats (no dominades) a $G_{cl}(t)$ es busca recursivament quin es el node pare fins arribar a s, per tal d'enregistrar quin es el camí corresponent a cadascuna de les possibilitats i retornar aquest conjunt de camins com a solució.
3. SELECCIÓ DEL CAMÍ. Seleccionar la possibilitat $(n, \vec{g_{ni}}, \vec{f_{ni}})$ de $OPEN$ amb major prioritat a la cua i no dominat a $G_{op}(n) \cup G_{cl}(n) \cup G_{cl}(t)$.

Eliminar $(n, \vec{g_{ni}}, \vec{f_{ni}})$ de $OPEN$ i moure aquesta possibilitat des de $G_{op}(n)$ a $G_{cl}(n)$.

Finalment filtrar de $G_{op}(n) \cup G_{cl}(n)$ totes les possibilitats dominades per la nova possibilitat $(n, \vec{g_{ni}}, \vec{f_{ni}})$.

4. EXPANSIÓ DEL CAMÍ. Si $n \neq t$, llavors: Per a tots els nodes successors m de n amb totes les possibles velocitats v_k:

   - Calcular el cost del nou camí trobat a m: $\vec{g_{mi}} = \vec{g_{ni}} + \vec{c}(n, m, t_i, v_k)$.
   - Calcular $\vec{f_{mi}}$ i comprovar no dominància respecte $G_{op}(n) \cup G_{cl}(n)$.
   - Si m es un node explorat:
     - Filtrar possibilitats dominades per $\vec{f_{mi}}$ en $G_{op}(m) \cup G_{cl}(m)$.
   - Inserir $(m, \vec{g_{mi}}, \vec{f_{mi}})$ a $OPEN$ i a $G_{op}(m)$.

   Fi del FOR. Retornar al pas de *COMPROVAR FINALITZACIÓ*.

8.4 Admissibilitat

Com hem mencionat en l'apartat d'admissibilitat de NAMOA*, un algoritme multi-objectiu es *admissible* si finalitza amb el conjunt de solucions no dominades entre si quant aquest conjunt es finit i no buit, o si no finalitza quant aquest conjunt es infinit. No obstant, en aquest algoritme no podem assegurar la seva admissibilitat, donat que s'ha dissenyat amb l'objectiu de resoldre un problema concret, i per tant certes adaptacions implementades fan que aquesta condició mencionada no es pugui assegurar. Com hem descrit abans, en aquest algoritme es prescindeix de les comprovacions de SG per evitar cicles per poder alliberar-nos del gran increment de temps de càlcul que això pressuposa. No obstant, donades unes condicions artificials extremes, es possible que l'algoritme es quedi bloquejat per el recorregut infinit d'un cicle de nodes que indiquessin cost zero (tot i que aquesta possibilitat es impossible a la pràctica (costos consecutius de distancia o de consum de combustible zero, per

exemple)). A més, certes adaptacions posteriors, com la modificació de la granularitat de l'altura d'ona(descrita mes endavant), propicien que el conjunt de solucions no dominades que proporciona l'algoritme no sigui exactament el conjunt total de solucions possibles (ja que aquesta adaptació elimina la redundància de solucions molt similars entre si).

8.5 Eficiència

Tal com hem vist a NAMOA*, es necessari que l'heurístic utilitzat sigui consistent. La propietat d'optimalitat en la poda de camins C*-limitats si es disposa d'un funció heurística consistent es segueix aplicant en aquest algoritme.

8.6 Optimalitat

En el cas d'aquest problema, els requeriments de temps i espai son encara mes grans que els de la cerca multi-objectiva descrita a NAMOA*, degut a la variabilitat dels costos de cadascuna de les caselles al llarg del temps, així com la utilització de més d'una velocitat. No obstant, degut al tractament específic d'aquestes dues característiques (descrit en altres apartats), la optimalitat del algoritme es aproximadament la mateixa que en NAMOA* (sense tindre en compte les comprovacions de SG a NAMOA*).

Capítol 9

Adaptació al problema

Hi ha diverses parts del problema inicial que s'han anat adaptant a mesura que hem anat implementant l'algoritme per així poder-ho adaptar als aspectes de la navegació marítima real, en aquest apartat descriurem els aspectes a adaptar més rellevants:

9.1 Visors

La part mes rellevant, que ja estava prevista al inici del projecte, es el fet de que si es vol arribar un usuari final, com es pretén amb el programa a implementar, no es poden introduir les dades i accedir als resultats de forma poc intuïtiva. Es per això que es necessari incloure un visualitzador que sigui usable per poder interactuar amb l'algoritme en el programa.

El visualitzador que estava previst per al projecte era un visualitzador molt senzill implementat amb les llibreries QT, ja que es prioritzava molt més de moment una bona eficiència algorísmica que no una interacció completa. El visualitzador implementat d'aquesta manera té aquest aspecte

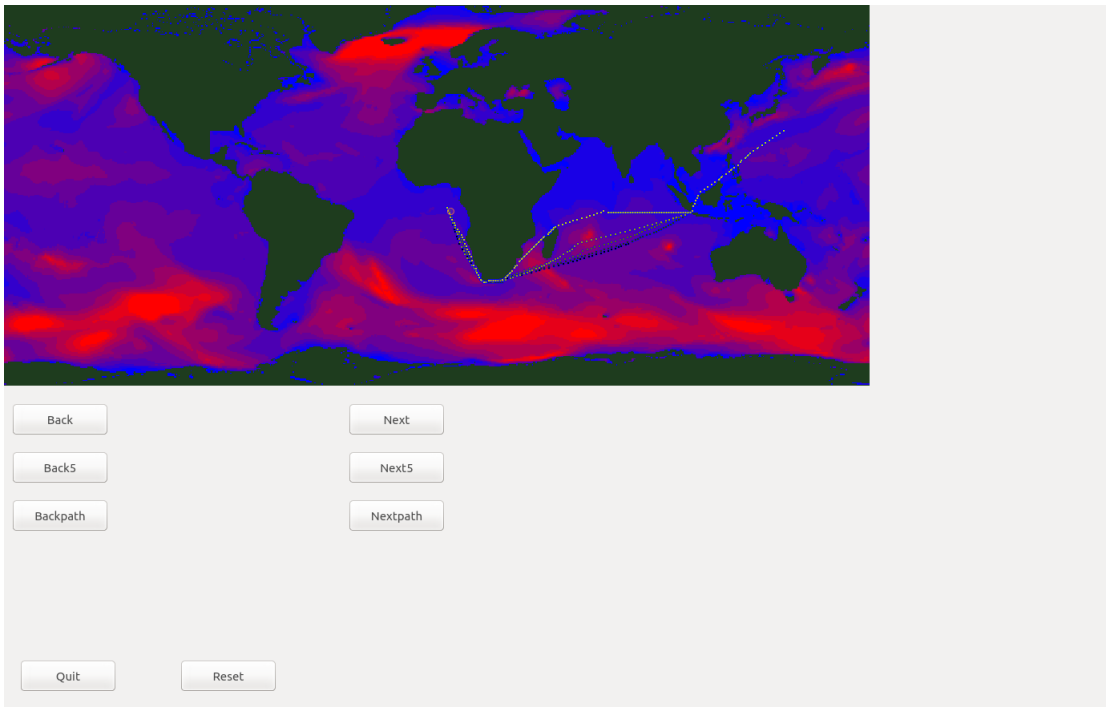


Figura 9.1: Primer visor basat en QT

Lo important d'aquest visualitzador es que es suficientment funcional, l'usuari tan sols ha de fer clic en dos punts navegables d'un mapa per a poder obtindre un resultat (el botons addicionals serveixen per veure el progrés de la meteorología al llarg del temps junt amb l'evolució de les rutes). No obstant, les dades addicionals sobre preferències de optimització (quins elements a optimitzar) s'han d'introduir en un arxiu de text a part, i les dades meteorològiques es tenen que actualitzar de forma manual. De totes formes, aquesta versió del visualitzador va acabar sent relegada a utilitzar-se tan sols per debugar quant encara estava en una versió molt bàsica, ja que, amb la continua interacció amb la empresa SIMO setmanalment, es va arribar a la conclusió de que com finalment el projecte es presentaria al abril seria convenient implementar un segon visualitzador molt més intuïtiu, ja que es disposava d'altres visualitzadors de la companyia per poder partir d'una base. Així que, amb la col·laboració de SIMO, es va desenvolupar un visualitzador molt més avançat utilitzant javascript i els mapes globals BingMaps. Podem veure el resultat en la següent imatge:

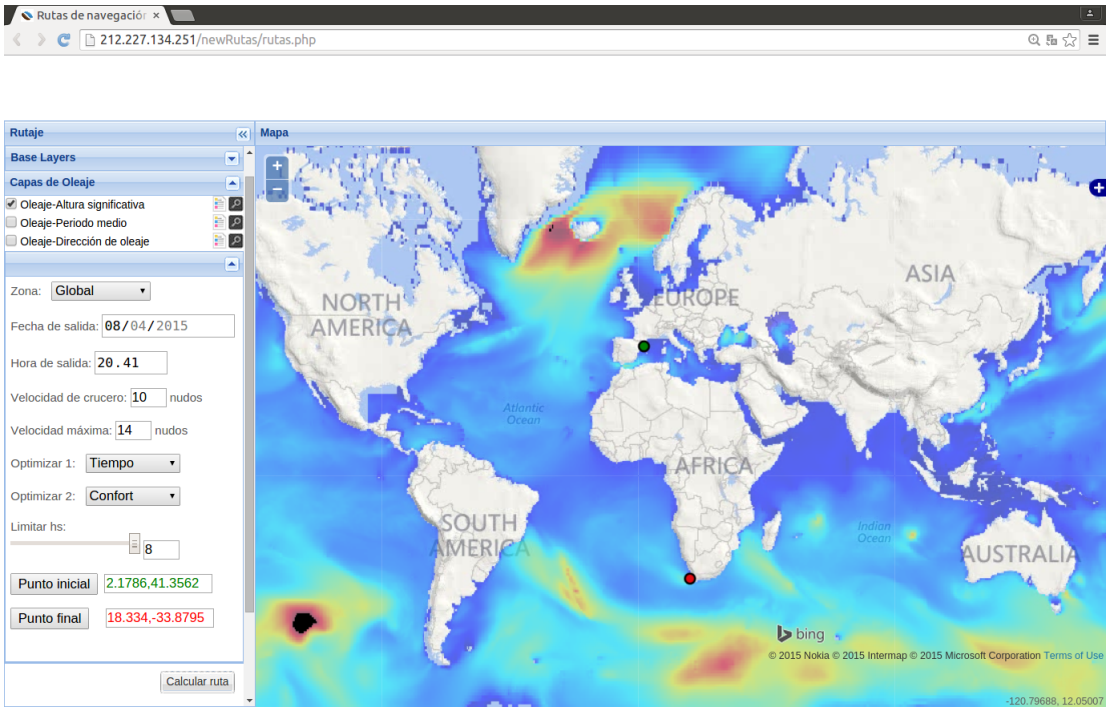


Figura 9.2: Segon visor basat en javascript i BingMaps

Com podem veure, el disseny es molt superior, les dades globals que utilitza son reals i detallades, i al ser basat en Bing Maps el visualitzador disposa d'una interacció molt més satisfactòria. Les dades meteorològiques s'actualitzen diàriament a través d'un script (ara mateix l'aplicació esta desada en un servidor extern), i l'usuari pot detallar les seves preferències de forma usable a la barra de l'esquerra (incloent-hi quin mapa dels tres vol utilitzar). A més, quant es mostra la solució, l'usuari pot fer un clic a un enllaç per accedir a una gràfica que compara les diferencies al llarg del temps del nivell d'altura d'ona (en l'apartat d'exemples de rutes es pot veure un exemple), i pot passar el ratolí per sobre de les rutes per veure les dades rellevants que conté cadascun dels nodes.

9.2 Distancies realistes

En un entorn artificial, la distancia entre dos grups de dos nodes separats pel mateix número de nodes seria la mateixa. No obstant, en aquest problema això no es cert, ja que tan sols es compliria si la terra no fos esfèrica. Per tant, la distancia entre dos nodes passara a ser la distancia ortodròmica. Aixi donç, definirem la distancia entre

dos nodes u_i i u_{i+1} amb latituds i longituds lat_i , lon_i , lat_{i+1} i lon_{i+1} respectivament de la següent manera:

$$\sqrt{(lat_i - lat_{i+1})^2 + ((lon_i - lon_{i+1}) * cos(lat_i + lat_{i+1}))^2}$$

Cal observar que aquesta fórmula ens retornarà el resultat en milles nàutiques, que son equivalents a 1852 metres cadascuna.

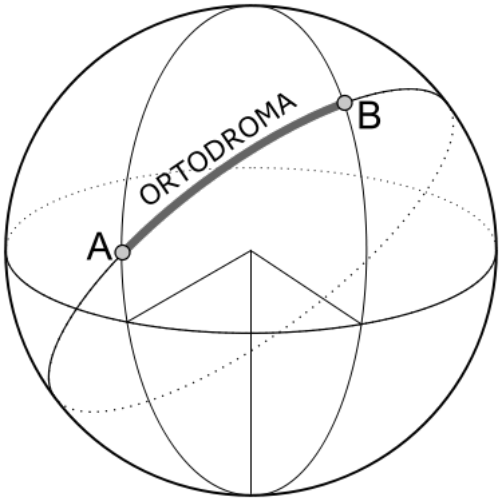


Figura 9.3: Descripció visual d'una línia ortodròmica entre dos punts

9.3 Obstacles entre nodes veïns

Hem definit els veïns d'un node com els 8 nodes adjaçents al mateix + 32 nodes arbitraris per poder simular canvis de rumb més suaus. No obstant, això genera un problema obvi, i es que si aquests nodes no son immediatament adjacents pot haver-hi obstacles (nodes definits com a terra) que impedeixin el pas. Per aquest motiu, donat un node inicial x_{ij} i un possible node veí $x_{i'j'}$ definirem un conjunt de nodes O_{ij} tals que, si cap d'ells es definit com a terra, el possible node veí $x_{i'j'}$ no es valid. Aquest seria un exemple dels possibles obstacles entre un node inicial i un node veí qualsevol:

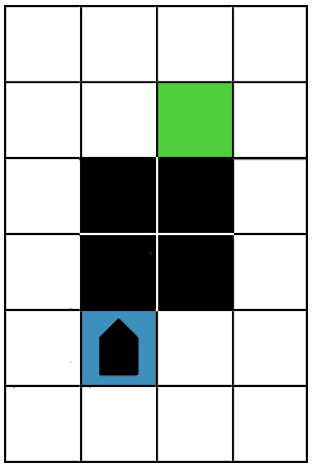


Figura 9.4: Possibles obstacles del recorregut entre dos veïns

9.4 Precisió respecte a la latitud i longitud inicials

En una situació real els punts inicials i finals d'un recorregut venen definits per les seves latituds i longituds respectives. Això vol dir que cal adaptar aquest factor als requisits del problema, es a dir, relacionar quin es el node inicial i el final que corresponen amb la descripció de longituds i latituds. No obstant, això no es plausible amb total precisió, ja que la diferencia de longitud o latitud entre un node i un altre adjacents horitzontalment o verticalment es de 0.5 graus al mapa global, 0.1 al mapa del mar mediterrani i 0.03333 al de Catalunya. Per tant, es necessari aplicar un mètode alternatiu que ens permeti realitzar els càlculs amb total precisió.

En aquest cas, sabem que, si busquem el node amb la latitud i longitud mes pròximes al punt desitjat, aquest punt es trobara **dintre del àrea** (o en el límit de la mateixa) que composa aquest node. Sent així, una primera solució es pressuposar que el node inicial/final es el node mes pròxim al punt de longitud i latitud donades. No obstant, aquest error pot ser de la meitat de la diagonal de la longitud d'un node, i en el cas sobretot del mapa global no es gaire recomanable. No obstant, la solució es simple, tan sols cal que en el primer i ultim recorregut entre dos nodes (es a dir, el transcurs entre n_0 i n_1, i entre n_{k-1} i n_k) calculem les distancies reals (entre el origen/destí i el centre del node següent/anterior, respectivament), i aixi no cal modificar el tractament que l'algoritme principal fa dels nodes, ja que si per exemple

estem optimitzant segons el temps, ja calculara quant de temps es triga entre els dos nodes segons la distancia real, i no la aproximació anteriorment esmentada.

9.5 Granularitat de les dades d'altura d'ona

Una vegada es va implementar el NAMOA* bàsic es va poder comprovar que anava molt lent inclús en les distancies mes petites del mediterrani. El motiu principal es que trobava una gran quantitat de camins a efectes pràctics redundants. Aquestos resultats eren deguts al fet de que dels dos paràmetres a optimitzar (temps i altura d'ona(confort)) no se'n havia fet cap preprocessament per adaptar-ho adequadament, per tal de que les rutes de les solucions fossin substancialment diferents, sent per exemple un camí qualsevol diferent d'un altre per +1 minut de temps i -1 cm d'altura d'ona. Aquesta imatge mostra un dels resultats obtinguts per una de les primeres versions del algoritme:

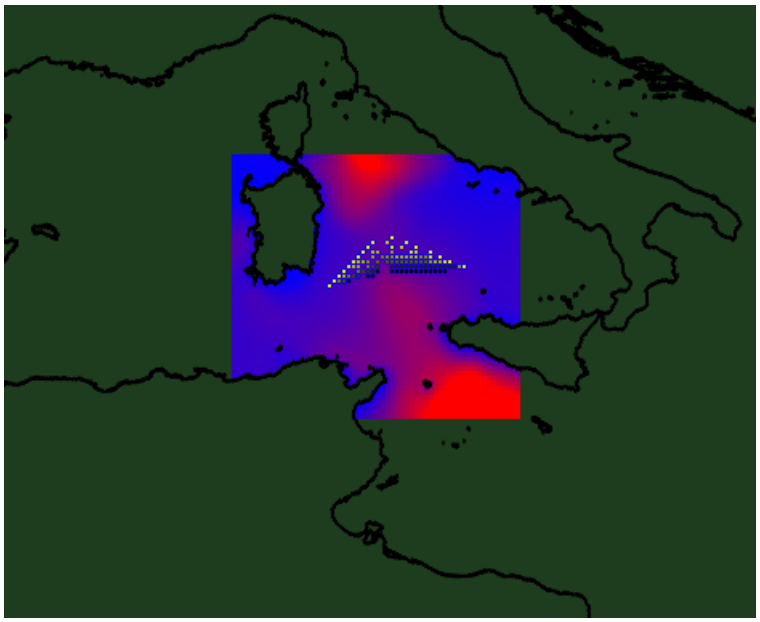


Figura 9.5: Granularitat d'altura d'ona alta

La solució no obstant es força senzilla una vegada trobada, tan sols cal modificar la granularitat d'un dels dos paràmetres (en el nostre cas l'altura d'ona, a nivells de 25 cm). Gràcies a aixo, vam poder obtindre uns resultats mes interessants, i a un temps de càlcul molt més baix. De fet, els resultats abans d'aquesta millora eren massa

costosos per ser considerats, una execució tan senzilla com la de la imatge anterior podria trigar minut i mig, i això limitant la regió a explorar a la zona quadrada que es veu. Per tant, el problema es simplifica molt quant reduïm la granularitat tal com mostra aquesta imatge d'un dels visors de l'algoritme una mica més avançat:

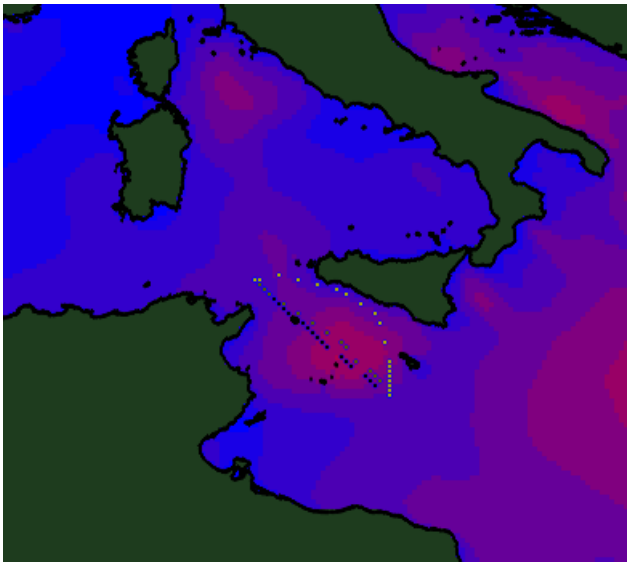


Figura 9.6: Granularitat d'altura d'ona baixa

9.6 Limit de les prediccions meteorològiques

Les prediccions meteorològiques que ofereix SIMO per al projecte son del paquet de previsions de PREVIMER. Aquestes previsions, si ve extenses, òbviament tenen els seus límits tant en espai (com hem mencionat en l'anterior punt), com sobretot, temporalment.

9.6.1 Limit de la capacitat predictiva

Com es lògic, tan sols es disposa d'un número limitat de temps en les prediccions, de fet les prediccions de PREVIMER normalment tenen una llargada de dies. Per tant, es fàcil veure que en els viatges que requereixin de més de 6 dies no disposarem de tota la informació necessària per realitzar la predicció. Per tant ens veiem obligats a treballar amb dades antigues a partir dels 6 dies. No obstant, aquesta no es una limitació molt seria, ja que en les circumstancies de que un viatge sigui d'una

setmana o més (o inclús si no es aixi) es normal que les prediccions es realitzin diàriament per part de la capitania d'una nau.

9.6.2 Granularitat de les mostres temporals

Les previsions s'emmagatzemen amb una mostra cada 3 hores. Per tant, la predicció de l'estat de la mar en l'interval entre les mateixes ens es desconegut. No obstant, degut a la naturalesa de la mar, es casi segur que l'estat de la mar en un moment donat s'assembli molt a l'estat anterior o posterior en el mateix node, o en el dels veïns. Per tant, una possible solució senzilla i eficient, que es la que aplicarem en el projecte, consisteix en simplement realitzar la interpolació entre les dos mostres del node mes properes temporalment de que es disposi.

9.7 Tractament de les diferents velocitats

Tal com hem descrit en la definició del algoritme, un aspecte clau es el tractament de les diferents velocitats per tal de mantenir el cost computacional en uns límits raonables. Paral·lelament, segons les descripcions de les companyies amb experiència en el sector de la navegació marítima, com SIMO, en un recorregut de distancia considerable, una nau intenta mantindre una velocitat de creuer el major temps possible, per tal de mantindre eficiència de combustible. Per tant, en el transcurs des d'un node n en temps t fins a un node m, disposant d'una funció MET que ens descriu la meteorologia d'un node qualsevol node n' en un temps qualsevol t', tan sols considerarem utilitzar una velocitat diferent a la de creuer si

$$MET(m, t_s + timecost(n, m, t_s, v_i)) \prec MET(m, t_s + timecost(n, m, t_s, v_{creuer}))$$

Aquest mètode per utilitzar diferents velocitats es molt senzill, però els resultats son satisfactoris segons experts en navegació marítima de SIMO, tot i que estan limitats tan sols a velocitats menors a la de creuer, degut a que si s'optimitza el temps de viatge amb una velocitat disponible superior a la de creuer es aquesta la utilitzada majoritàriament degut a la ordenació de les possibilitats a $OPEN$. En

el futur s'haurà de buscar una solució per aquesta limitació, degut a que disposar d'una velocitat superior també pot ajudar a evitar meteorologia adversa.

Capítol 10

Exemples de rutes

En aquesta secció mostrarem alguns exemples del funcionament del algoritme en rutes conegudes. Per fer-ho adjuntarem en cadascuna de les rutes una imatge del resultat final del visor (on es pot veure el mapa, amb un fons que descriu l'altura d'ona en el moment inicial i les rutes de la solució), i un altra imatge mostrant la progressió dels dos elements a optimitzar (altura d'ona (**confort**) al llarg del **temps** del recorregut) i finalment inclourem el temps de calcul necessari per trobar la solució final. S'ha de tindre en compte que com més rutes i més gran el número de nodes a explorar, més probable es que el calcul d'una solució es compliqui. Totes aquestes execucions s'han realitzat amb una maquina de les següents característiques:

- *Processador*: Intel Core i7-4650U
- *Nuclis de CPU*: 4 Cores x 1.7 GHz
- *Memoria RAM*: 8 GB de RAM DDR3
- *Disc dur*: 512 GB SSD

10.1 Mapa de Catalunya

El mapa de Catalunya es el mapa amb menys nodes de tots els que disposem, i no disposa de gaires obstacles, per lo que les seves solucions son força fàcils d'obtindre (no obstant, la granularitat es mes alta, aixi que el detall de les solucions fa que no sigui trivial). Les gràfiques son les generades automàticament pel visor i ens podem fixar que la ruta amb el pic d'altura d'ona màxim mes baix, es sempre el que tarda mes en arribar al node destinació.

10.1.1 Barcelona - Ajaccio

Temps d'execució: 0.548324 segons

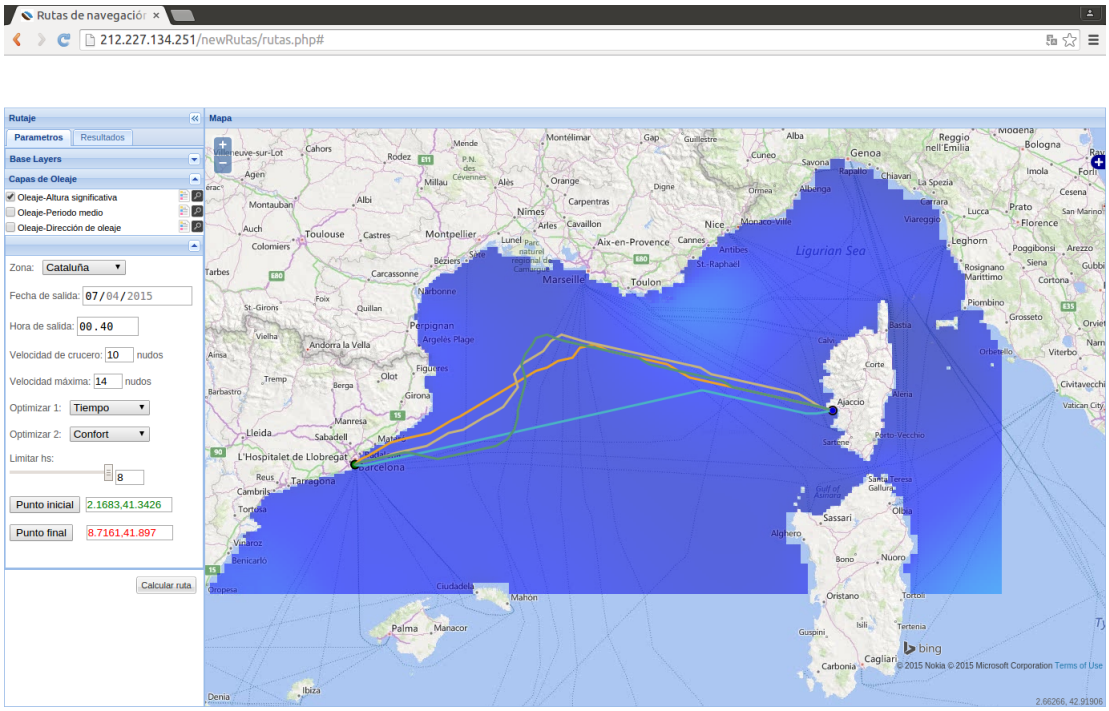


Figura 10.1: Bacelona-Ajaccio

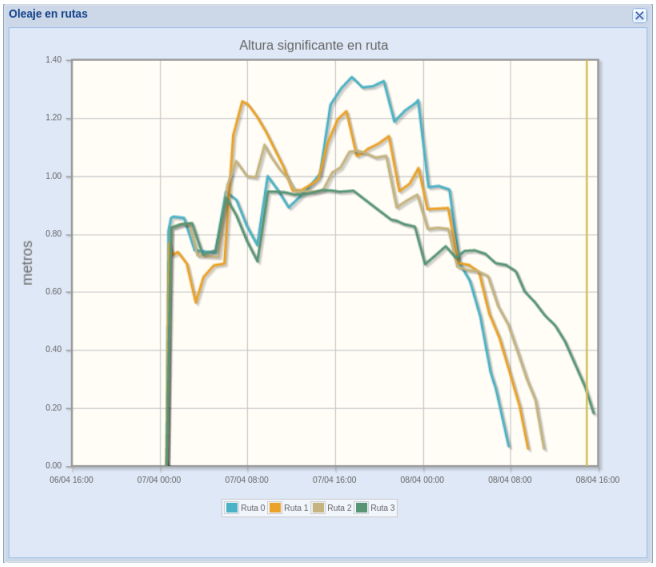


Figura 10.2: Gràfica Bacelona-Ajaccio

10.2 Mapa del mar mediterrani

En un principi el projecte anava a ser tan sols sobre el mapa mediterrani. Aquest mapa ja té casos en els que trobar solució no es del tot fàcil, i en anteriors versions

podia ser problemàtic, però en la versió actual funciona satisfactòriament.

10.2.1 Barcelona - Civitavecchia

Temps d'execució: 0.701443 segons

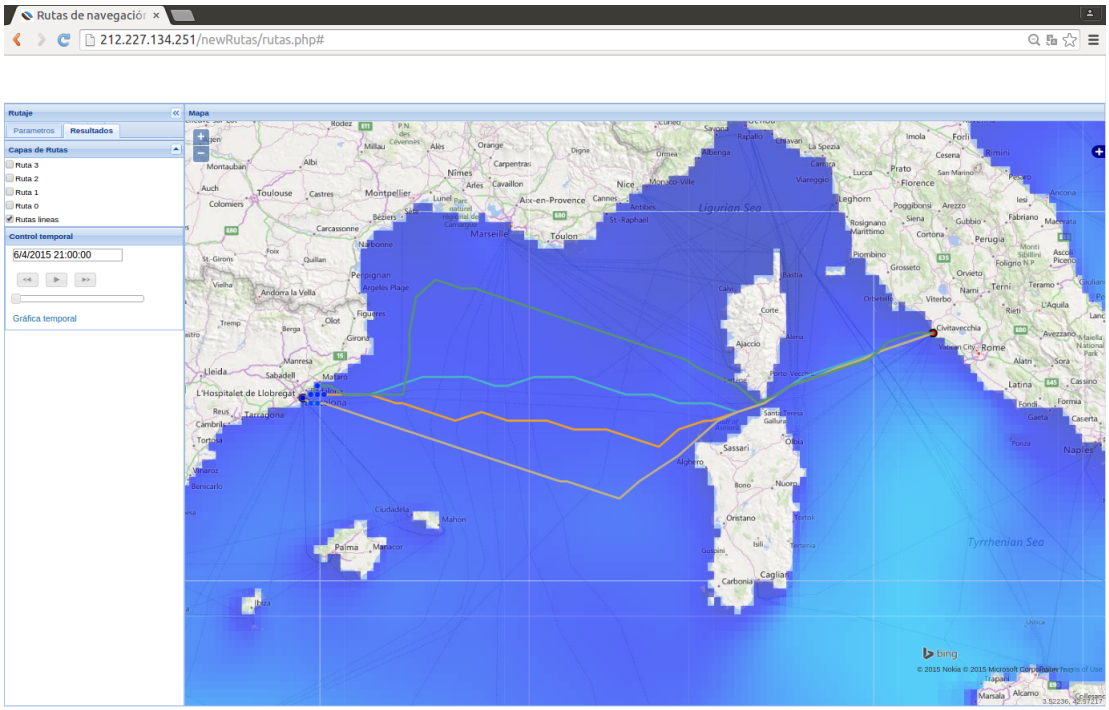


Figura 10.3: Bacelona-Civitavecchia

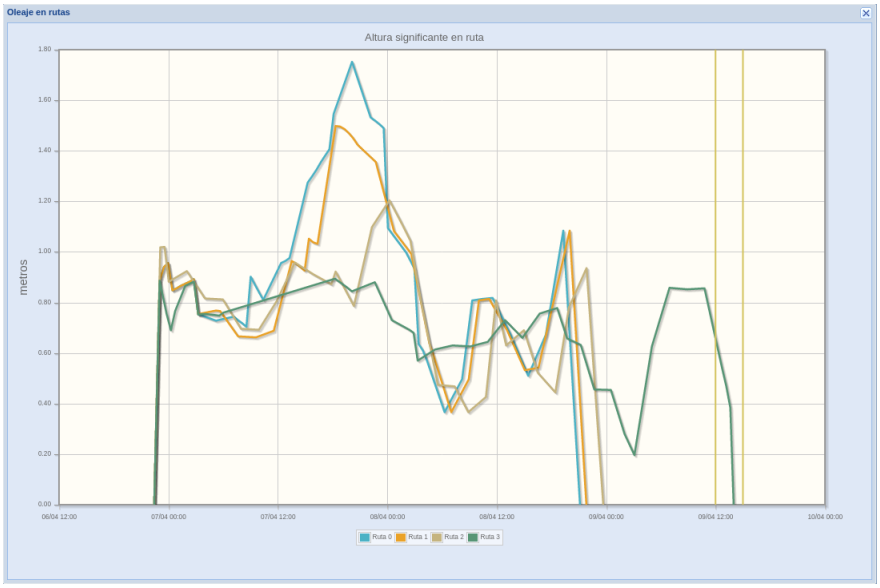


Figura 10.4: Gràfica Bacelona-Civitavecchia

10.2.2 Barcelona - Atenes

Temps d'execució: 2.07128 segons

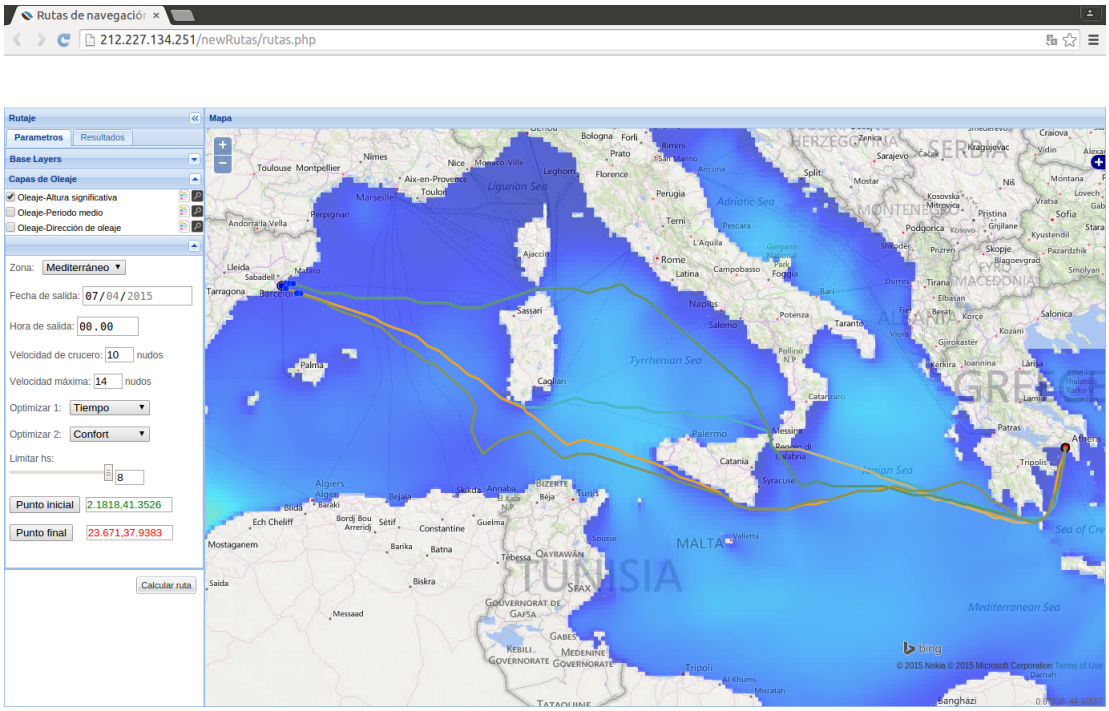


Figura 10.5: Bacelona-Atenes

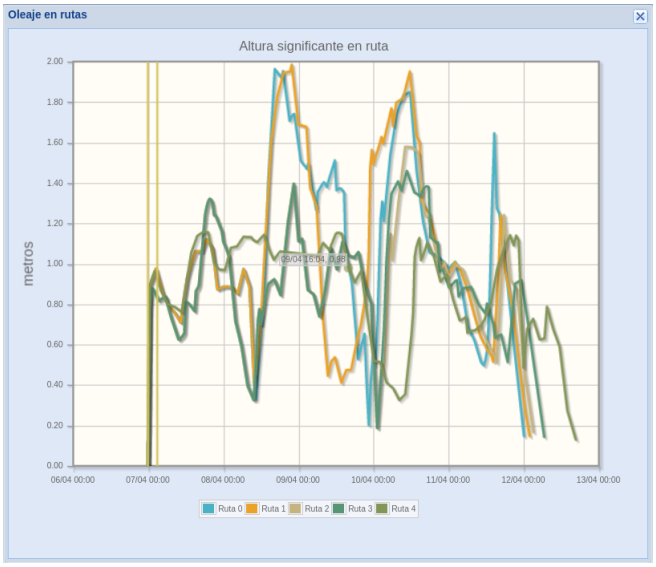


Figura 10.6: Gràfica Bacelona-Atenes

10.3 Mapa global

Aquest mapa ja presenta situacions difícils, i es el que conté els casos extrems del programa degut a la seva gran magnitud. S'ha de tindre en compte que aquest mapa es l'únic en el que els limits est i oest no son infranquejables, ja que es passa directament a l'altra extrem horitzontal del mapa, tal com podem veure a l'ultima

ruta (on hem tret el fons d'altura significativa per a que es puguin veure bé els camins).

10.3.1 Londres - Nova York

Temps d'execució: 1.63172 segons

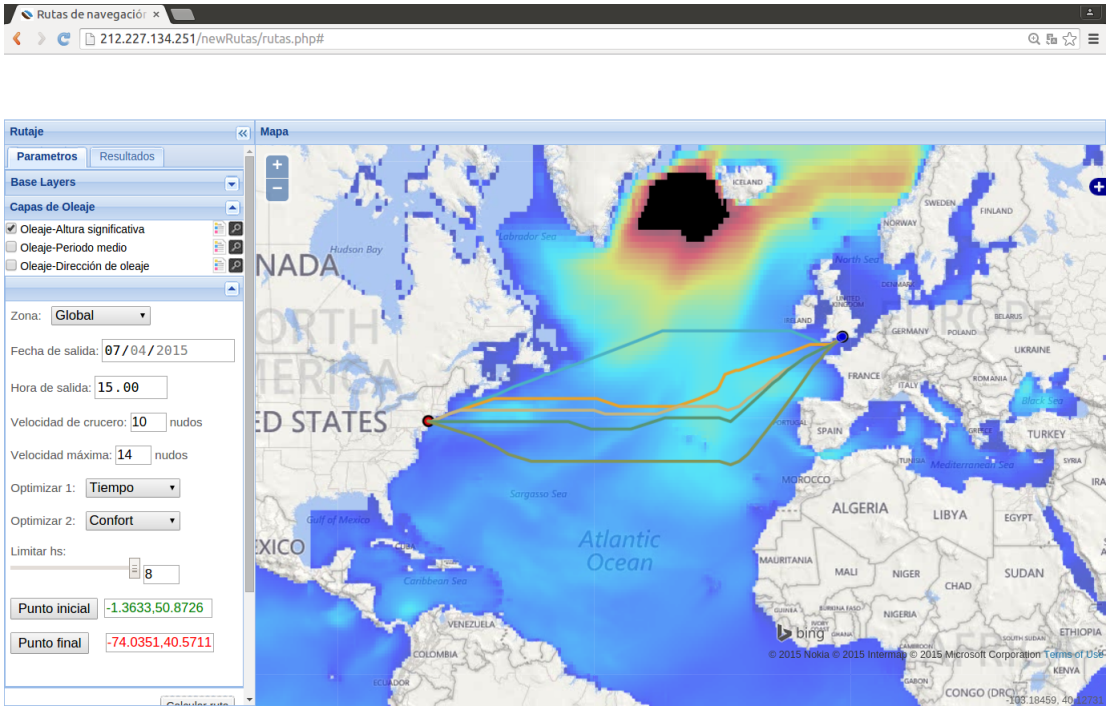


Figura 10.7: Londres-Nova York

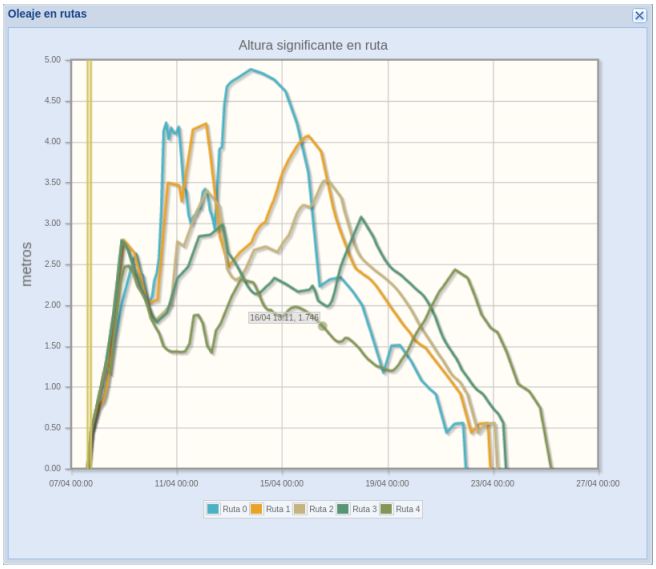


Figura 10.8: Gràfica Londres-Nova York

10.3.2 Barcelona - Kuala Lumpur

Temps d'execució: 4.49752 segons

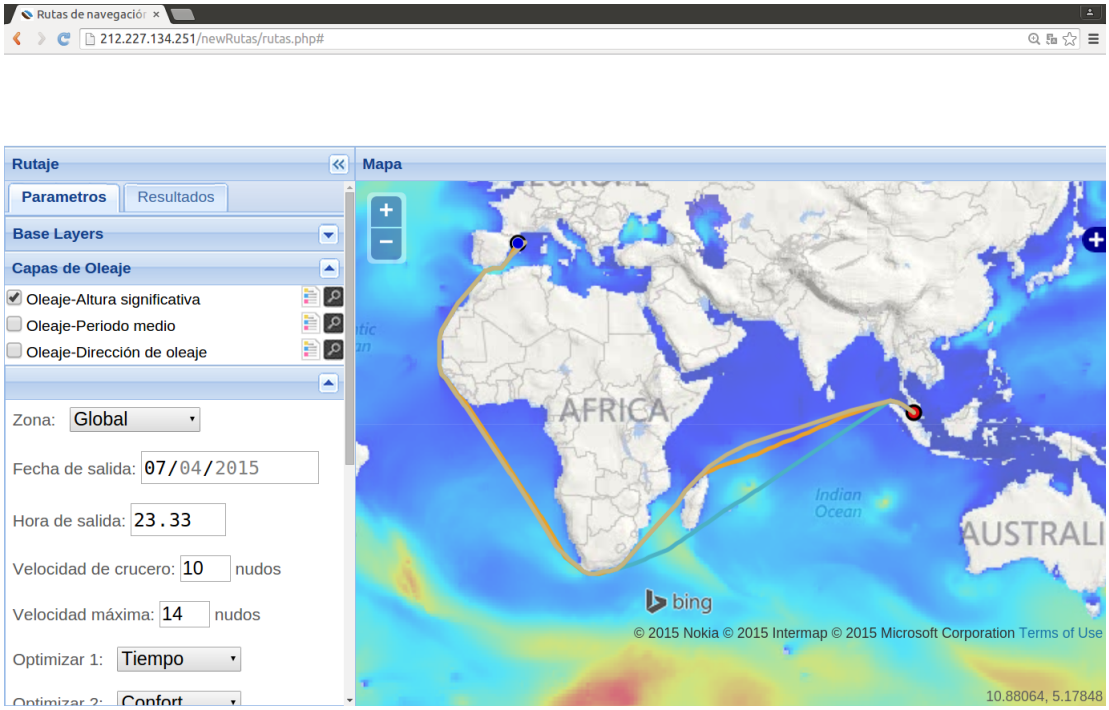


Figura 10.9: Barcelona-Kuala Lumpur

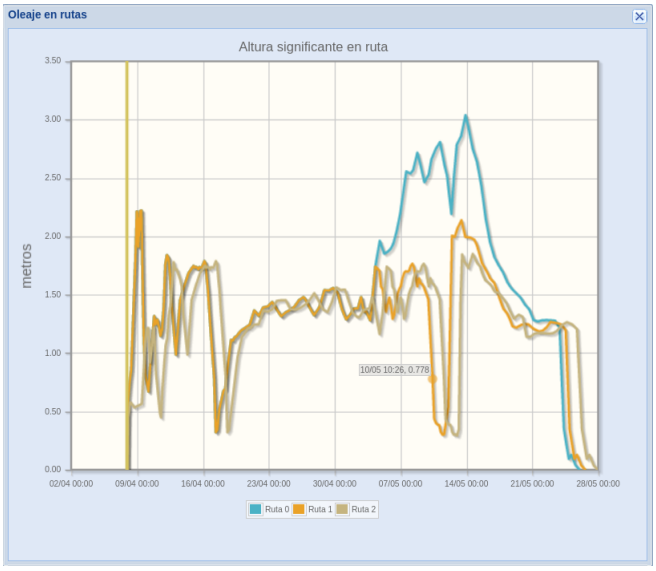


Figura 10.10: Gràfica Barcelona-Kuala Lumpur

10.3.3 Panamà - Toquio

Temps d'execució: 4.72244 segons

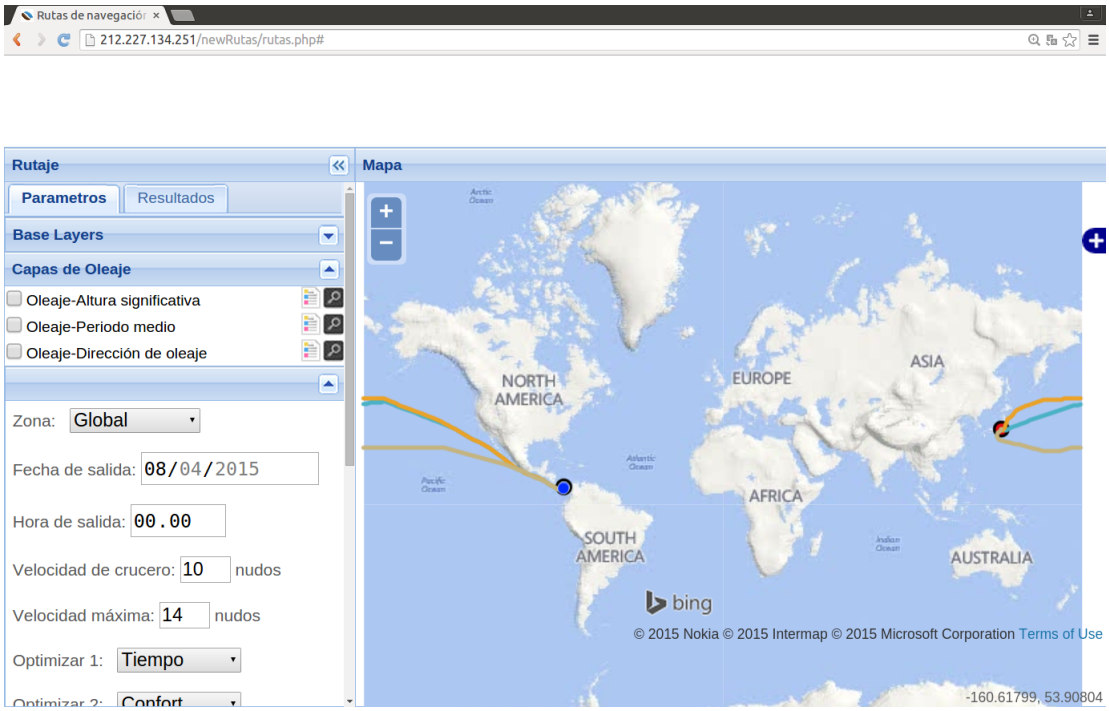


Figura 10.11: Panamà-Tòquio

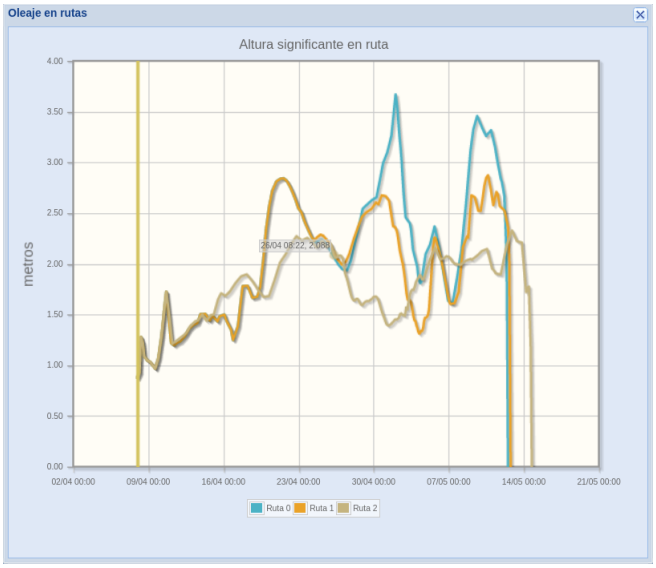


Figura 10.12: Gràfica Panamà-Tòquio

Capítol 11

Millores futures i conclusions

11.1 Millores futures

Hi han diverses millores futures possibles que serien interessants per l'algoritme:

- **Velocitat superior a la de creuer**. La velocitat de creuer es la velocitat bàsica en la que realitzen la bona part del recorregut la majoria de naus. Com hem mencionat abans una de les millores mes immediates seria resoldre el problema de poder utilitzar una velocitat superior a la de creuer sense influir en gran mesura en el temps de calcul. Com en les velocitats inferiors a la de creuer, aquesta no s'espera ser utilitzada més que en situacions molt específiques (com per exemple quant es pugui passar a través d'una tempesta abans de que es formi del tot). Una solució possible es un tractament posterior de la solució considerant aquestes velocitats, però esta per veure si una solució aixi resol el problema satisfactòriament.

- **Màxima precisió disponible**. La diferencia principal de les característiques dels mapes disponibles es que com mes gran sigui un mapa pitjor serà la seva precisió en les dades. No obstant, l'usuari final espera com més precisió millor, i per tant seria bona idea en els mapes mes generals (com per exemple el mapa global) modificar l'algoritme per a que utilitzés les dades mes precises de les que disposi en cada zona (per exemple en el mapa global en la zona de Catalunya utilitzar les dades del mapa de Catalunya). No obstant, aquesta no es una modificació trivial, ja que augmentaria força el temps de calcul i crear una malla asimètrica pot ser difícil. La solució podria ser executar l'algoritme un cop al mapa mes general, i després executar iterativament l'algoritme per cada

mapa menor amb punts inici destí limitats per aquell mapa (per exemple, en la ruta Barcelona-Nova York, una vegada executat l'algoritme global, es podria tornar a calcular per al mapa mediterrani per cada ruta trobada, amb punt d'inici Barcelona i destinació l'ultim node de cada camí abans de sortir del mapa del mar mediterrani (en aquest cas l'estret de Gibraltar)).

- **Dades específiques de les naus**. Es molt important saber quin tipus de nau està sol·licitant el recorregut per poder adaptar-lo a les sever característiques. Per exemple hi han naus que no poden passar per certs estrets, o que es veuen molt més afectades per aspectes de la meteorologia com el vent (els velers) o aspectes marítims com el període de l'onatge (creuers). És per això que en un futur es pot desenvolupar una base de dades amb les dades de cada vaixell dels usuaris per així poder adaptar el els càlculs a cadascun d'ells.

- **Mapa de costes detallat**. Les dades que utilitzem per a l'algoritme estan aproximades, això vol dir que en cada mapa, n metres quadrats estan representats per un sol node, i mapes molt generals com el global poden donar solucions que no considerin aquest nivell de baix detall (com per exemple no tenint en compte una illa molt petita i proposant alguna ruta que passi a través d'ella). Es per això que en el futur es important introduir aquestes comprovacions mitjançant algun mapa de costes addicional. No obstant aquestes comprovacions serien segurament massa costoses d'introduir al algoritme principal així que lo millor seria probablement aplicar un ajustament posterior a les rutes de la solució.

11.2 Conclusions

El programa resultant d'aquest projecte resol de forma adequada el problema proposat. Les parts interessades, tant el director del projecte com SIMO, troben les solucions obtingudes amb el programa com a satisfactòries, i el visor posterior desenvolupat (junt amb SIMO) proporciona una bona interacció amb l'usuari.

L'eficiència de l'algoritme ha sigut inclús major de l'esperada gràcies a les adaptacions de l'algorisme al problema a resoldre, com per exemple l'eliminació del graf acíclic SG o la reducció en la granularitat de l'altura d'ona.
Encara queda treball futur per millorar l'aplicació actual, i depenent de l'interès futur pel programa es pot avançar i perfilar cap a un producte comercial real destinat a usuaris del sector.

Finalment, personalment aquest projecte ha suposat una experiència molt interessant, sent un bon exemple realista d'aplicació d'un algoritme avançat a un problema molt real, i el fet de poder col·laborar amb un expert en computer science com és el director del projecte, i una empresa tant pro-activa i innovadora en el sector com és SIMO ha fet que aquesta experiència hagi sigut molt enriquidora tant a nivell acadèmic com a nivell professional.

Part IV

Bibliografia i glossari

Bibliografia

[1] Lawrence Mandow and José. Luis Pérez De La Cruz. Multiobjective a* search with consistent heuristics. *J. ACM*, 57(5):27:1–27:25, June 2008. 8, 9, 10, 41, 42

[2] Alexander Schrijver. On the history of the shortest path problem. *Documenta Math. 155*, 2010. 13

[3] A. Shimbel. Structure in communication nets. *Proceedings of the Symposium on Information Networks*, pages 199–203, 1955. 13

[4] L. Rosenfeld. Unusual problems and their solutions by digital computer techniques. pages 79–82, 1956. 13

[5] Richard Bellman. On a routing problem. *Quart. Appl. Math.*, 16:87–90, 1958. 13

[6] Lester R. Ford Jr. Network flow theory. *RAND Corporation*, Paper P-923, 1956. 13

[7] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959. 13, 20

[8] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems, Science, and Cybernetics*, SSC-4(2):100–107, 1968. 14, 36

[9] Richard E. Korf. Depth-first iterative-deepening: An optimal admissible tree search. *Artif. Intell.*, 27(1):97–109, September 1985. 14

[10] Richard E. Korf. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial Intelligence*, 27:97–109, 1985. 14

[11] R. Motte. *Weather routeing of ships.* Maritime Press, 1972. 15

[12] N. Bowditch. American practical navigation. *National Imagery and Mapping Agency, Bethesda.*, 1995. 15

[13] Hoffschildt, Bidlot M., Hansen J. R., and Janssen B. Potential benefit of ensemble forecasts for shiprouting. *ECMWF Technical Memorandum no. 287.*, 1999. 15

[14] O. Saetra. Ensemble shiprouting. *ECMWF Technical Memorandum no. 435.*, 2004. 15

[15] C. U. Böttner. Weather routing for ships in degraded conditions. *International Symposium on Safety. Security and Environmental Protection.*, 2007. 15

[16] Rina Dechter and Judea Pearl. Generalized best-first search strategies and the optimality of a*. *J. ACM*, 32(3):505–536, July 1985. 37

[17] Bradley S. Stewart and Chelsea C. White, III. Multiobjective a*. *J. ACM*, 38(4):775–814, October 1991. 40, 42

[18] L. Mandow and J. L. Pérez de la Cruz. Comparison of heuristics in multiobjective a* search. In *Proceedings of the 11th Spanish Association Conference on Current Topics in Artificial Intelligence*, CAEPIA'05, pages 180–189, Berlin, Heidelberg, 2006. Springer-Verlag. 41

[19] J. L. Bentley, H. T. Kung, M. Schkolnick, and C. D. Thompson. On the average number of maxima in a set of vectors and applications. *J. ACM*, 25(4):536–543, October 1978. 42

[20] Brian Logan and Natasha Alechina. A* with bounded costs. In *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, AAAI '98/IAAI '98, pages 444–449, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence.

[21] Lawrence Mandow and José. Luis Pérez De La Cruz. Multicriteria heuristic search. *Europ. J. Operat. Res*, pages 253–280, 2003.

Glossari d'abreviatures

NAMOA* New Approach to Multi-Objective A*. 3, 8–10, 12, 20, 37, 40–45, 47, 48

NetCDF Network Common Data Form. 26, 28

PREVIMER PRÉVIsion de la MER. 25, 26, 28, 55

SIMO Soluciones Ingeniería Marítima Operacional. 12, 25–28, 50, 55, 56, 67

TXT Tecnologies per a Tothom. 27

WERMED WEatherRouting dans la MÉDiterranée. 5, 16, 17

WMS Web Map Service. 28